FORM

IST-1999-10357

Engineering a Co-operative Inter-Enterprise Management Framework Supporting Dynamic Federated Organisations Management

Document Number:	IST-1999-10357/TCD/WP3/012
Title of Deliverable:	D12 – FORM Methodology for the Open Development Framework
Deliverable Type: (P/R/L/I)*	P
Nature of the Deliverable: (P/R/S/T/O)**	R
Contractual Date of Delivery to the CEC:	28-02-2002
Actual Date of Delivery to the CEC:	28-02-2002

Workpackage responsible for the Deliverable:	WP3/4
Editor:	Vincent Wade (TCD)
Contributor(s):	D Lewis (UCL)
Reviewer(s):	B Cullen (TCD), C Harris (TCD)

ABSTRACT

This document describes the FORM Methodology for development of Building Blocks and Management Systems using the Open Development Framework. The Methodology comprises two Guidelines – namely the Building Block Development Guideline and the Business Process Driven System Development Guideline, which target different kinds of development and hence can be used by different types of developer.

The *Building Block Development Guideline* describes how Building Blocks for network, service and system management can be developed for the FORM Framework. This guideline is intended for use by Management Component Developers/Vendors and, where new functionality is required, Management System Integrators. The *Business Process Driven System Development Guideline* provides guidance for the construction of management solutions that implement management business processes using FORM Building Blocks Contracts. This guideline is intended for use by Management System Integrators and Manager Service Providers

KEYWORDS

Methodology, Development Process, Business Process, Software Component,

* Type: P:Public, R-Restricted, L-Limited, I-Internal
** Nature: P-Prototype, R-Report, S-Specification, T-Tool, O-Other

IST-1999-10357

FORM

Deliverable D12:

FORM Development Methodology for the Open Development Framework

Editor :	Vincent P. Wade
Status – Version :	Version 2
Date :	2002-02-28
Distribution :	Public
Code :	IST-1999-10357/TCD/WP3/012

The FORM Consortium consists of:

Atos Origin Intégration (France) – *Project Coordinator*

Broadcom Eireann Research Ltd. (Ireland)	Trinity College Dublin (Ireland)
DELTA Danish, Electronics, Light & Acoustics (Denmark)	UH Communications A/S (Denmark)
Fraunhofer FOKUS (Germany)	University College London (UK)
LM Ericsson A/S (Denmark)	Waterford Institute of Technology (Ireland)
TDC Tele Danmark A/S (Denmark)	KPN Research (The Netherlands)

Table of Contents

Executive Summary

This document describes the Development Methodology portion of the FORM Open Development Framework (ODF). The ODF Methodology aims to support the construction of management systems from separately sourced software components. The ODF Methodology addresses the need for a common understanding of the models representing products exchanged by different types of stakeholders in a potential market for management components. At the same time the ODF Methodology aims to support the convergence of the stakeholders' development processes with best practice in software development. The main types of artifacts produced by the ODF Methodology are:

- Off the shelf software components that are developed by Independent Software Vendors for use by System Integrators. Within the ODF such components are termed *Building Blocks*.
- *Management Systems* that are assembled from Building Blocks by System Integrators for use by Service Providers.

The Methodology comprises two Guidelines – namely the Building Block Development Guideline and the Business Process Driven System Development Guideline, which target different kinds of development and hence can be used by different types of developer.

The ***Building Block Development Guideline*** describes how Building Blocks for network, service and system management can be developed for the FORM Framework. Each Building Block represents a manageable, re-usable, deployable unit of management functionality. In order to provide greater flexibility for implementation the abstract concept of a Building Block Contract as the means for interacting with a Building Block is supported by the Guideline. The Building Block Contract specifies a grouping of information and behaviours, which can be re-used to support management business processes. One or more Building Block Contracts can be supported (implemented) by a Building Block in the FORM Framework. This guideline is intended for use by Management Component Developers/Vendors and, where new functionality is required, Management System Integrators. This guideline is presented as a related set of development stages and workflows. The guideline prescribes UML modeling notations and models to characterize the key stages in the development process. The Guideline also prescribes a XML based description of Building Block Contracts. This guideline is loosely based on the Rational Unified Process.

The ***Business Process Driven System Development Guideline*** provides guidance for the construction of management solutions that implement management business processes using FORM Building Blocks Contracts. This guideline is intended for use by Management System Integrators and Manager Service Providers. This guideline also utilises UML modeling notations.

1. Introduction

As a result of deregulation of the telecommunications industry in Europe and the US, there has been a major growth in the number and sophistication of telecommunication services. These services range from virtual private networks to value added services such as collaborative working environments. Because of the increased competition, telecommunication operators are under increasing pressure to accelerate the deployment and provisioning of telecommunications services. Rapid and flexible management automation is seen as a key competitive discriminator between operators. Moreover, because of the cost and complexity of building bespoke telecommunication service management systems, telecommunications providers are moving towards the use of off the shelf components to satisfy their management requirements. These key business drivers have been responsible for the increased interest in new approaches to the rapid and flexible development of component-based telecommunications service management systems.

This document describes the Development Methodology portion of the FORM Open Development Framework (ODF). The ODF aims to support the construction of management systems from separately sourced software components. The ODF addresses the need for a common understanding of the models representing products exchanged by different types of stakeholders in a potential market for management components. At the same time the ODF aims to support the convergence of the stakeholders' development processes with industry best practice.

Motivation for Methodology

The methodology tackles the twin challenges of designing reusable components and providing a component-based approach to the implementation of business process driven management systems. The Methodology and Framework are being validated within the context of developing B2C and B2B management solutions although they can be applied to a wider scope of communications services management markets. In tackling the challenge of managing B2B and B2C services, the different stakeholders involved in the supply chain of B2B managed services are recognised – namely the producers of open interface standards, the providers of off-the-shelf component software, the developers of management systems which use those components and the service providers who operate those systems. The requirements of these stakeholders vary but all must be accommodated by the methodology. For instance, Service Providers and Management System Integrators require that management solutions can be rapidly constructed and flexibly deployed at low cost through the reuse of software components. Component providers wish to supply this need in a manner that supports as wide a reuse market as possible.

The Methodology proposes a business process driven approach to the construction of management systems solutions from re-usable software components, which are termed Building Blocks (BB). However, management component providers need to be able to add new Building Blocks to the framework or enhance existing building blocks in the

Framework. Such development activity requires guidance concerning the development and specification of Building Blocks. Therefore the FORM Methodology proposes a guideline to support the development of management Building Blocks. Thus the FORM Development Methodology is divided into two separate but consistent guidelines, namely:

1. The ***Building Block Development Guideline***: This guideline is intended for management component providers developing management Building Blocks for reuse by management service providers or management system integrators.
2. The ***Business Process Driven System Development Guideline***: This is intended for use by system integrators who are developing management systems, based on business process analysis techniques, and who wish to use off the shelf management building blocks.

.

2. Background: the FORM Open Development Framework

The Guidelines presented in this document are part of an Open Development Framework (ODF) developed by the FORM project. The ODF provides the business context and requirements for which the Development Methodologies have been designed. The following subsection provides an overview of the ODF and of its constituent Logical Architecture that provides the meta-model and principles that guided the design of the Methodological Guidelines.

2.1 Overview of ODF

The FORM project proposes that a common framework is required to develop and exploit open interfaces in developing component-based management software. This is termed to Open Development Framework (ODF).

The ODF has the goal of supporting the construction of management systems from separately sourced software components. To help refine this goal a set of stakeholders has been defined which represent the principle organisation types involved in developing management systems from components. They are as follows.

- Standards Bodies, which produce the industry agreements that underpin interoperability and integration of separately sourced software components.
- Independent Software Vendors (ISVs), which produce and market software components.
- System Integrators, which produce management system constructed from separately sourced software components, including ones that are developed internally.
- Service Providers, which possess the business requirements for management systems and operate them.

There may be occasions where an organisation takes on more than one the roles represented by these stakeholder types, e.g. a large service provider may produce standards for procurement purposes and may have internal division producing management system or components. However, these stakeholders provide a relatively simple but comprehensive model for the market interactions the ODF must address. These interactions include:

- The consumption of standard specifications by Service Providers for procurement, by System Integrators and ISVs for software development and by other Standards bodies.
- The consumption of software components from one or more ISVs, as well as those developed internally, for management systems development by System Integrators.
- The procurement by a Service Provider from a System Integrator of a management system that must integrate with management systems from other sources, already present in its own OSS and those of other Service Providers.

The scope of the ODF is further refined by the following aims:

1. Support common mechanisms for the communication of products (both spec-ware and software) between these stakeholders
2. Ensure the processes for developing the products exchanged between stakeholders converge with industry best practice such as model-based development and the use of UML.
3. Support the management of products once they have been made available by a stakeholder in order to support their later reuse by others.
4. Encourage the separation of techniques for integrating between different technologies from those for integrating between different models.

The ODF does not explicitly aim to support automatic model checking or code generation, but does aim to exploit the current capabilities of widespread modeling tools. Additionally, it is not an aim to endorse any particular technology, though guidance is given in the issues related to technology selection.

The ODF is structured into four portions: a Logical Architecture, a Development Methodology, a Technology Architecture and a Set of Reusable Elements. This breakdown follows a pattern observable in other management related frameworks, e.g. ITU-T’s Telecommunications Management Network architecture, the OMG’s Open Management Architecture etc.

The concerns addressed by the four portions of the Framework are:

- **Logical Architecture:** The Logical Architecture describes the structural concepts of the ODF and their relationships. This is described in terms of a meta-model that is used to ensure consistency of the models generated when applying the guidelines present in the rest of the ODF. The core structural concept is the software Building Block (BB), which is an atomic unit of software deployment and management. A BB implements a number of Contracts that are the sole medium via which BB interact with their environment. Management systems are built primarily from assemblies of BBs. Systems can be modelled at business level in terms of Business Processes and Business Roles. Reference Points exist between Business Roles and are realised through Contract implementations. To promote their reuse, Contracts may be described in a technology neutral format which can be transformed to one or more technology specific versions for implementation in BBs. Contract definitions also include specifications of the information passed via the Contract by reference to an explicit information model.

- **Development Methodology:** The Development Methodology describes the processes and notations needed to design Contracts, develop Building Blocks and assemble management systems that conform to the ODF. The primary modeling notation used is UML, though XML is used for packaging UML with other model formats in Contract and BB specifications. The methodology integrates a number of existing modeling techniques such as use case modeling, business process modeling and model-view-controller analysis modeling plus the variety of other modeling approaches supported by UML. The Methodology contains two Guidelines, one for the development of Building Blocks and the other for the development of business processes based management systems that are mostly assembled from of Building Blocks.
- **Technology Architecture:** The Technology Architecture addresses how the concepts expressed in the Logical Architecture can be implemented using individual technologies. It diverges from the technology related portion of most other frameworks in that it does not attempt to promote the use of a specific technology or integration technique. Instead, it addresses the issues to be considered when making a technology selection for a specific Management System, set of Contracts or group of BBs. This includes issues related to functional capabilities of technologies (e.g. data handling capabilities, security, transaction support) and non-functional capabilities (e.g. availability, scalability).
- **Reusable Elements:** This portion of the ODF is the repository for reusable products that result when the ODF is applied to a particular application domain, e.g. the IES Management domain addressed in FORM. The principle types of reusable entities are: Business Role definitions; Reference Point specifications; Contract specifications and their grouping into BB Specifications and BB implementations. Other types of reusable elements, such as policy and business process definitions are also being investigated.

2.2 Overview of Logical Architecture

The Logical Architecture defines the architectural principles that underpin the rest of the ODF. It also defines an architectural meta-model that defines the various models to be used when applying the ODF and the relationships between those models that must be maintained to reap the full benefits of the ODF. These architectural principles and the architectural meta-model form the prescriptive part of the Logical Architecture. The Logical Architecture also contains some descriptive elements included to help in the understanding of the prescriptive parts and their application. These descriptive elements include the definition of a set of abstract roles representing potential users of the ODF at different points in the lifecycle of its main structural elements. This approach is taken because a key feature of a component-based architecture is that it addresses a broad range of the software development lifecycle and therefore brings together concerns that are only relevant to certain roles in the overall software lifecycle. These user roles are used to describe the different way in which the model elements defined in the meta-model would be used and exchanged by the ODF stakeholders.

The architectural principles are based on an analysis of the state-of-the-art in management software architectures and component based development, in particular the NGOSS architecture being developed by the TM Forum. These principles can be summarised as follows:

- The core component structure is modelled as a Building Block that offers one or more interfaces called Contracts.
- Building Blocks and Contracts are packaged with business requirements and system analysis models.
- The modeling of a Contract may be performed independently of implementation technologies, including the explicit modeling of the information that passed by a Contract[1].
- The modeling of Building Blocks and Contracts should support the separate expression of any Business Logic that may be configured after implementation.

The full description of these principles is accompanied by the associated benefits each principle imparts to the different ODF stakeholders.

The main structural concepts presented in the Framework are grouped in terms of the models generated by the ODF user roles and thus used in exchanging information between those roles. These models are:

- Business Context Model: This captures the requirements for an area of concern, modeling its organisation environment and defines a business-level model of its externally observable functionality and internal business processes.
- Domain Analysis Model: This analyses an area of concern, defining a system analysis level model of its externally observable functionality, process behaviour and logical decomposition.
- Contract Set Specification: This is a set of interface specifications that may be utilised in designing BB software. It provides the primary means of interoperability by detailing sets of interfaces that can be used in interacting with Building Blocks.
- External Information Model: This is a technology neutral expression of the information that is potentially passed via a set of Contracts. This provides the primary mechanism for ensuring interoperability between Contracts defined in different technologies.
- Building Block Group: This is a set of Building Block implementations and its accompanying documentation.
- Management System Model: This is the description of a Management System's design, which uses Building Blocks to some extent.

The relationships between these models are depicted in the following figure.

[1] Currently the information models of the Contracts are defined in a technology neutral form. However, the interface functions or message structures are technology dependent.

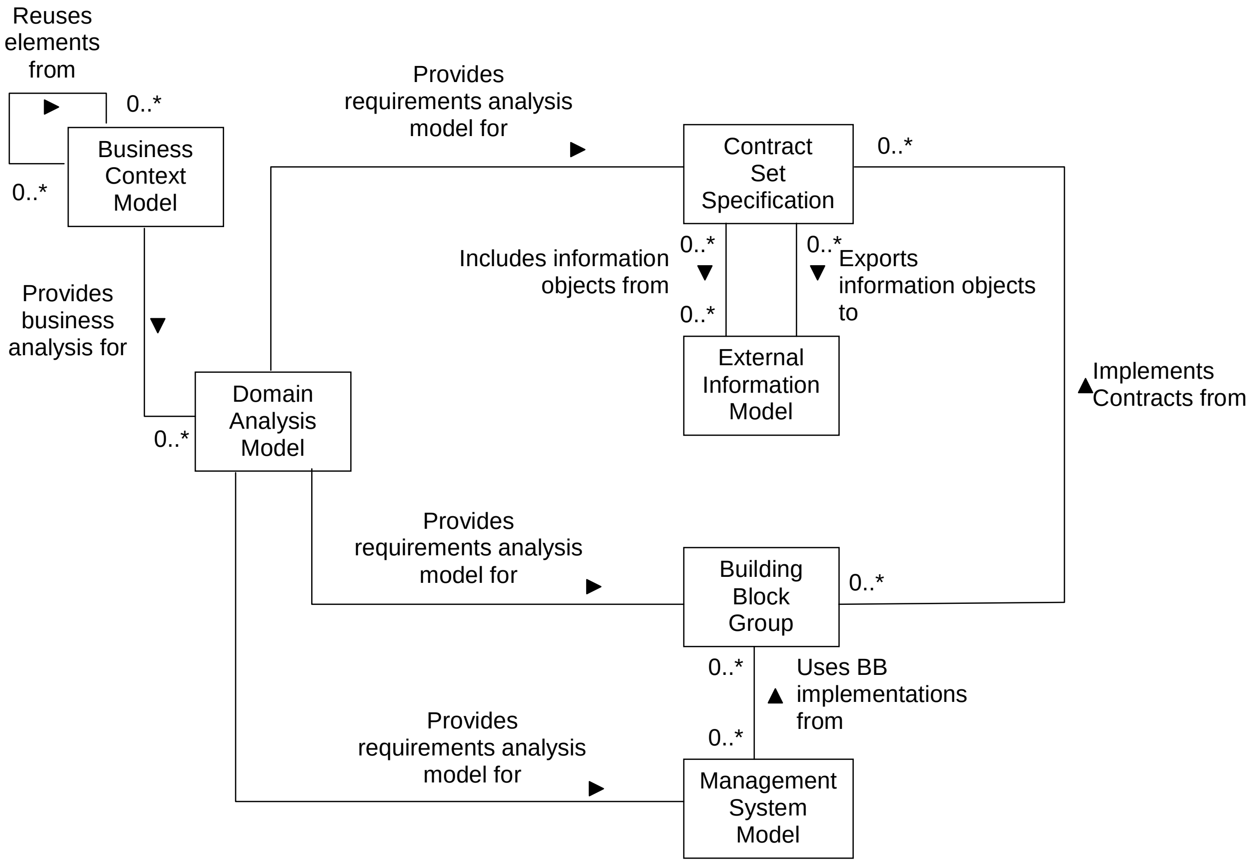

Reuses elements from
0..*
0..*
Business Context Model
Provides business analysis for
0..*
Domain Analysis Model
Provides requirements analysis model for
Contract Set Specification
0..*
Includes information objects from
0..*
0..*
0..*
Exports information objects to
External Information Model
Implements Contracts from
Provides requirements analysis model for
Building Block Group
0..*
0..*
Uses BB implementations from
0..*
Provides requirements analysis model for
Management System Model

Figure 2-1: High-level mate model for ODF Logical Architecture

Details of the Logical Architecture, it architectural principles and meta-model together with its mapping to modeling artifacts used in the Guidelines presented in this document can be found in FORM deliverable D9. The Guidelines presented in this document address the generation of models covering most of the elements of the Logical Architecture's meta-model. This table below identifies the ODF meta-model elements, the development workflows which guide their development, and the UML representations which capture those elements:

ODF Meta-Model Elements	Building Block Development Guideline Workflows	Business Process Driven System Development Guideline Workflows
Business Context Model	Developed by the *Perform Business Modeling Workflow* and *Define Reference Architecture Workflow*	Developed by the *Perform Business Modeling (and Reference Architecture Refinement Workflow)*
Domain Analysis Model	Developed by the *Define Requirements Analysis Model Workflow* and the *Develop*	Developed by *Define Requirements Analysis* in combination with *Model*

	Analysis Models Workflow. Used in *Reorganise Analysis Model Workflow*	*System Processes and Information Workflow*.
Contract Set Specification	Constituent Contracts developed as part of the *Reorganise Analysis Model Workflow*	Used in *Re-Model System Processes and Map to Building Back Contracts Workflow*
External Information Model	Developed as part of the *Reorganise Analysis Model Workflow*	Used in *Re-Model System Processes and Map to Building Back Contracts Workflow*
Building Block Group	Developed as part of the *Reorganise Analysis Model Workflow and Implement Building Blocks Workflow*	Used in *Map Building Block Contracts to Building Blocks and Define Component's Control and Data Flow Workflow*
Management System Model	Not applicable	Developed as part of *Model Missing Objects and Information Workflow*, and as part of *Implement Integration and Map Building Block Contracts to Building Blocks*

3. The ODF Development Methodology

The ODF methodology encapsulates the twin objectives of the Building Block Development Guideline and Management system construction using existing component. These objectives address the needs of those developing Building Blocks as well as those developing management systems based on business process modelling but constructed using building blocks. These objectives are elaborated in more detail in the following subsections.

3.1 Objectives and Scope of Building Block Development Guideline

The objective of Building Block Development Guideline is the development of re-usable management Building Blocks Contracts and Building Blocks. The Guideline not only provides advice as to how to model Building Blocks Contracts, but also prescribes how such Building Blocks Contracts should be represented so as to ensure that the contracts could be reusable by other actors (i.e. actors not involved in the development of the Building Block Contract).

More specifically, the objectives of the guidelines are to:

- Guide the design activities in developing Building Blocks Contracts & Building Blocks.
- Specify the development workflows required to design the Building Block Contract.

- Identify modeling notations and the models to be developed during each development workflow. Indicate the tracability of artifacts developed across the development workflows.
- Prescribe sets of artifacts[2] to characterize and communicate usage of Building Block Contracts.

The Guideline focuses on Model Driven Development. Thus the workflows defined in the guidelines focus on the modeling of UML artifacts and models necessary to capture the design of Building Blocks and Building Block Contracts. It is not within the scope of the guideline to extend the UML v1.4 standard. The guideline attempts to work within this UML specification in determining the development workflows and prescribing the appropriate UML model specifications which characterise BB Contracts and BBs.

The guideline focuses exclusively on the developmental workflows rather than project management workflows or environment development workflows. Also, since the guidelines concentrates on the modeling aspects of BB and BB Contract development, it does not detail the programming/coding aspects of development or technology specific aspects (i.e. technology implementation decision to use EJB rather than CORBA etc.), and detailed technology testing execution.

3.2 Objectives and Scope of Business Process Driven System Development Guideline

The objectives of this Business Process Development Guideline are

- To provide support for a ‘Business Process Driven’ approach to management system construction from re-usable Building Blocks Contracts
- To provide a development guideline which will allow management systems integrators to construct management solutions from Building Block Contracts.

The Guideline assumes the existence of catalogue(s) of Building Block Contract Specifications and the Building Blocks which support them. (It is expected that these catalogues can be generated using the Building Block Development Guideline). The catalogues are expected to have a functional overlap with the management solutions to be developed. Building Block Contract descriptions in the catalogues are conformant to the Contract Description template defined in the Building Block Development Guideline. Thus for each Building Block Contract, there are prescribed set of models and a description template for describing a Contract.

The Guideline provides a development methodology from business modeling to system testing. The typical starting point for the guideline is a Management System Integrator wishing to implement business process(es) using the FORM Framework.

As with the BB development guideline, the Business Process Driven System Development Guideline focuses on Model Driven Development. Thus the workflows

[2] An artifact is a piece of information that is created, changed and used by actors when performing development activities. An artifact can be a model, a model element or a document [Jacobson 2000].

defined in this guidelines focus on the modeling of UML artifacts and models necessary to capture the design of management systems constructed from Building Blocks and Building Block Contracts. The guideline does not attempt to extend the UML v1.4 standard. The guideline attempts to work within this UML specification in determining the development workflows and prescribing the appropriate UML model specifications which characterize the intended system and its construction.

The guideline focuses exclusively on the developmental workflows rather than project management workflows or environment development workflows. It does not detail the programming/coding aspects of development or technology specific aspects (i.e. technology implementation decision to use EJB rather than CORBA etc.), and detailed technology testing execution. However, the guideline enables the generation of XMI descriptions of the control (business) logic which captures the rules necessary for building block (and interface) integration. These XMI descriptions can be used as a basis for automated integration using a variety of technologies and approaches e.g. workflow engines, scripts.

4. Building Block Development Guideline

The approach taken in developing the Building Block Development Guideline was to use best practice in software development and add new workflows, model, artifacts and specifications to capture the necessary information for Building Block development. The Building Block Development Process is loosely based on the Rational Unified Process (RUP). Several of the RUP development workflows are generally applicable to software design e.g. business modeling, use case modeling etc. However, although generally useful, RUP does not support key modeling artifacts and design activities, which are fundamental to the guideline including the notion of Building Block Contract, management reference points. Appendix A provides a description of RUP, and a description of the relationship between the Building Block Development Guideline and the RUP development process.

The Building Block Development Guideline focuses on model driven development, and consists of two phases, namely Context Modeling and Building Block Development. In the Context Modeling phase, the objective is to model the overall vision within which the Building Blocks are expected to reside. The workflows in this phase focus on the overall business modeling (some of which the building blocks would ultimately support), requirements engineering, object analysis and design. The second phase focuses on the re-organisation of these models and designs as Building Blocks and Building Block Contracts. This phase produces the models, artifacts and specifications needed to capture the design of the Building Block Contracts and the Building Blocks which support them.

4.1 Building Block Development Guideline: Context Modelling Phase

In this phase of the Guideline focuses on establishing the boundaries of the domain addressed by the Building Block development effort. This involves creating 'vision' document(s) that outline the scope of the management processes to be addressed by the Building Blocks (and their contracts) under development. The vision document is key to the development work as it is the clear statement of the context and operational constraints of the domains in which the Building Blocks are envisioned.

An initial reference architecture is then devised for the Framework. This phase also involves the development of Business Model(s) and the initial Business Use Cases for the management process areas being investigated. The choice and scope of such management processes can be influenced by 'standardised' management processes e.g. TeleManagement Forum's Telecom Operation Map. For example, if the envisaged building blocks and contracts were to be related to the TeleManagement Forum's standardisation effort then these processes may reflect some of the activities concerned with their prescribed Fulfilment, Assurance and Billing processes. The selection and customisation of such management process descriptions is dependent on the process areas of the Building Blocks to be developed. Also during this phase, terminology and other relevant standards/models need to be identified and either influence on the emergent design identified e.g. IPDR for Accounting, DMTF CIM and IEFT QoS information models for Assurance.

An important outcome of this phase is the initial identification of potential Building Blocks.

Key outcomes of the this stage are:

- Vision Document(s) indicating the scope, context and management business process areas of interest.
- Business Models identifying the Business Roles e.g. organisation(s) and Actors and where the management business processes reside.
- Initial Reference Architecture with reference points and domain boundaries.
- Requirements Capture & Management Document.
- Use Cases for the chosen management business processes. In addition to specifying the use cases, activity models representing the control and data flow involved in each use case can be modelled if required.
- Analysis Object Collaboration Model from which candidate Building Blocks can be identified.
- Plan of how development work will proceed.

Note: As the Guideline focuses on the development of Building Blocks for Network and Service Management, it will use the term 'management processes' to identify specific related functional areas rather than 'management business process'. The term 'business processes' would also be appropriate but is less specific for the subject of this Guideline.

4.2 Process Workflows in the Context Modelling Phase

One of the fundamental challenges in developing Building Blocks for inter-enterprise management is to identify the appropriate aggregation of functionality, information resources and control logic. It is a huge task and a great challenge to attempt to develop such aggregations (i.e. Building Blocks and Building Block Contracts) from the outset. In fact, such 'bottom up' style development of Building Blocks is only possible where the functional domain has already been clearly represented and is well understood or where there is pre-existing detailed models and designs available. This is not necessarily the situation for the Open Development Framework, as research and development effort into the construction of such re-usable Building Blocks within the varied domain of B2B management is still limited. A top-down approach is suggested in this Guideline in order to identify the range of candidate process areas, information and control objects required for the Framework. Once the domain has been analysed, the actual analysis & design of the Building Blocks can begin. Where extensive pre-existing design models exist, the context modeling phase can be used to refine the existing models and identify where additional modeling is necessary.

This Guideline adopts a Business Model/Use Case Driven approach to represent the organisational domains and process areas of interest. Examples of organisation boundaries may be that of various actors in the world of Internet and application services e.g. Application Service Provider, Internet Service Provider, and Customer. Examples of Management Process Areas would include Fulfilment, Assurance and Billing Processes. The development activities and workflows to be carried out in the Context Modeling Phase are identified below.

1. **Perform Business Modeling Workflow –** This process workflow facilitates the definition of business model(s) based upon management business processes. This involves identifying Business Roles, Business information entities, Business Use Cases and Organisation Units.
2. **Define Reference Architecture Workflow** - Develop a Reference Architecture that identifies reference points between organisational boundaries, the placement of process areas within these boundaries and the relationships between these process areas across organisational boundaries.
3. **Perform Requirements Analysis Workflow –** This involves such development activities as Perform Requirements Analysis, Development of Use Cases and Supplementary Requirements Specification. As mentioned previously, this can, where required, involve the modeling of activity graphs (diagrams) to represent the various control and data flows in the use cases.
4. **Develop Analysis Object Models Workflow –** This involves the development of analysis objects and development of analysis collaboration models. This is supported by a customisation of the RUP Analysis & Design workflow. These analysis models are used in the Building Block Development Phase for identifying candidate Building Blocks.

These activities and workflows are described in detail below. Examples of models and artifacts developed using the methodology are also presented to illustrate the guideline's usage. These examples are drawn from the area of service assurance in FORM.

4.2.1 Perform Business Modelling Workflow

This workflow seeks to define the Business Roles, Business Use Cases, Business Processes and Business Resources/Information Entities with which the Actors interact. There are two aspects from which the Business Modelling is performed. The first is the External View of the Business, the second being the Internal View of the system. To model Management (Business) Processes the following models/diagrams are specified

(i) Business Use Case Diagram(s): depicting business roles (workers and/or organisations), use case name (external view)

(ii) Use Case Realisation, which models the business workers and entities/resources needed to carryout the use case (internal view)

(iii) Activity Diagram depicting the activities involved in carrying out the use case (internal view).

Figure 4.3 illustrates the necessary development activities defined in this workflow.

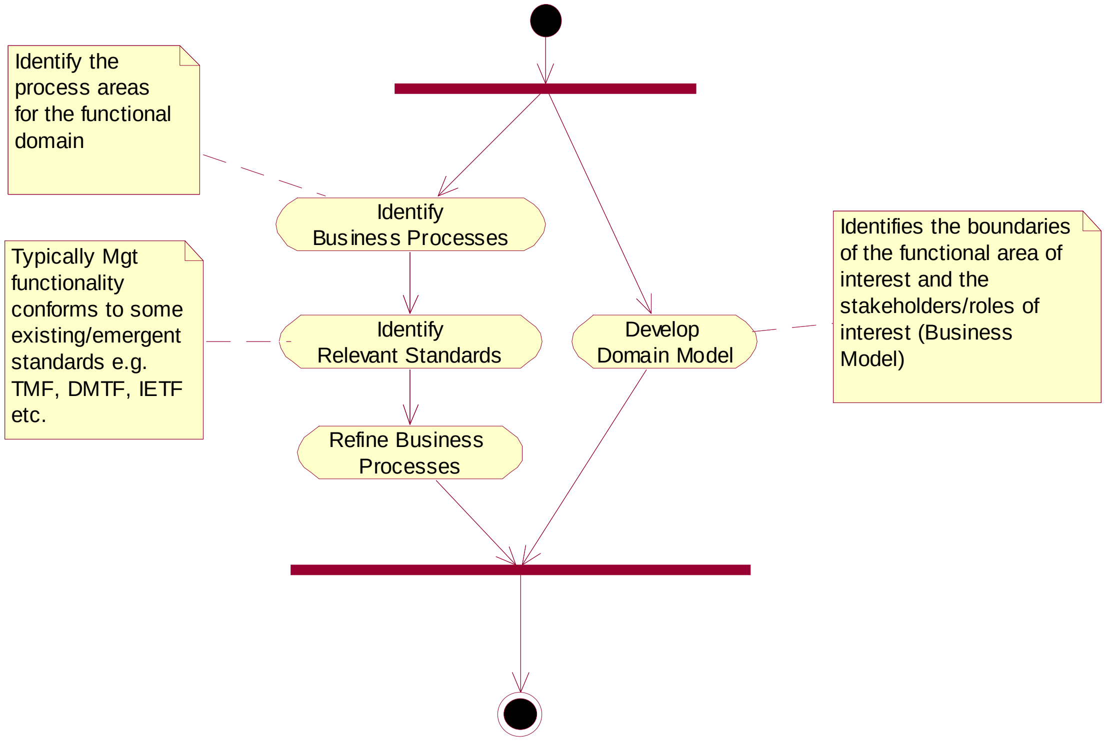

Identify the process areas for the functional domain
Identify Business Processes
Typically Mgt functionality conforms to some existing/emergent standards e.g. TMF, DMTF, IETF etc.
Identify Relevant Standards
Develop Domain Model
Identifies the boundaries of the functional area of interest and the stakeholders/roles of interest (Business Model)
Refine Business Processes

Figure 4.3 The Perform Business Modeling Workflow

4.2.2 Example Building Block Business Modelling Workflow

In order to better illustrate the development process, a case study will be presented, which outlines the development activities for each of the workflows. The example is concerned with the development of ‘Assurance’ building block(s). Suppose we wish to develop one or more building blocks and building block contracts to support the assurance of a WWW based information service. More specifically suppose this assurance set of building blocks are required to assure the end-to-end operation of the WWW based service (from information source to end consumer). Also suppose this end-to-end service assurance potentially involves the consumer of the service being connected via one or possibly multiple Internet Service Providers. The scope of these building block(s) however, is that of monitoring and managing the Web based information service and not the underlying network connectivity. Finally, suppose the this assurance service, may be offered by the Web based service provider, or by an independent service provider (called an Assurance Provider).

The first workflow (Perform Business Modeling) assists the designer in identifying the business context in which the building block(s) are expected to perform. The workflow prescribes the modeling of the Business Actors, Potential Business Organisation(s), and a set of use cases which would be required.

Figure 4 depicts the Business Model, where the independent organization (called the Inter Enterprise Service Provider or IESP) is providing the B2B service assurance between two enterprises (an Application Service Provider and its Customer Organisation). The business model also shows the assumption of a Network Assurance role within the IESP. This role is responsible for dealing with intermediary Internet Service Providers to provide quality assured IP connectivity service.

In the model the principle roles within the various organizations are represented as ‘business workers’. The relationships between these business workers is also represented to show, for example, the customer role being able to get a reporting service from the IESP.

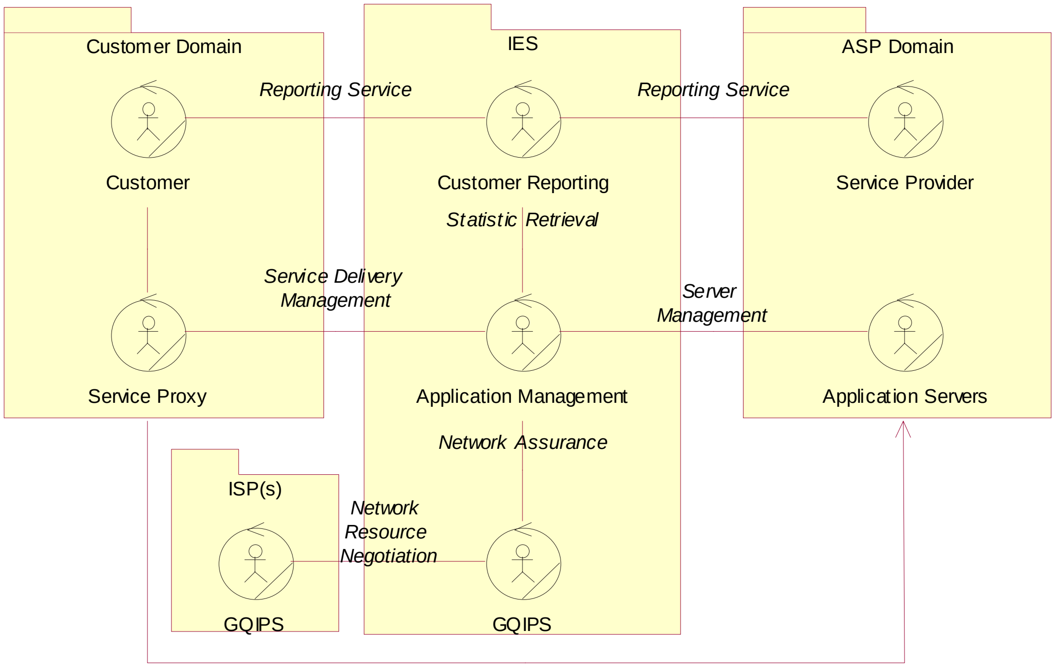


Figure 4.4 Business Model from Assurance

From this Business Model, actors and roles, an initial set of use cases can be defined. These define the services which are being offered with regard to assurance of the WWW based information. The use cases for this assurance service are presented in Figure 4.5

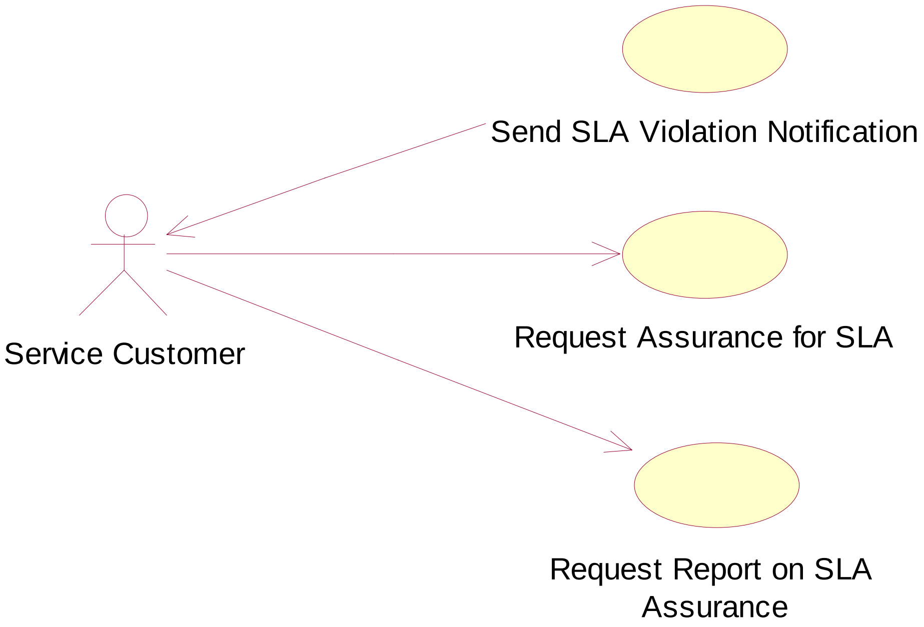

Send SLA Violation Notification
Service Customer
Request Assurance for SLA
Request Report on SLA Assurance

Figure 4.5 Business Use Case for Assurance

Thus the artifacts developed in this workflow are, Business Actors, Business Model and Business Use Cases. It is also possible to perform further business modeling by beginning to model the Business Activity Diagrams. These activity diagrams can be used as one of the inputs in the business functional requirements specification.

Also identified at this stage are the set of relevant standards bodies and their specifications e.g. DMTF & IETF standards for Assurance, TM Forum, IPDR for Accounting. In particular the TMForum has identified several business process areas, such as Assurance, Fulfilment and Billing. These process descriptions can be used either as a basis for the use cases & business activity diagrams, or just as a reference of typical telecommunications operator process requirements.

4.2.3 Define Reference Architecture Workflow

In order to provide a coherent logical structure through which the management business processes can integrate, a logical architecture is developed. It is useful to provide such a single diagram, which shows the logical separation of management processes and the candidate reference points between those management processes. This helps communicating/explaining how the functional areas could co-operate, as well as providing a common ‘map’ around which the development teams can co-ordinate. The notion of ‘reference point’ is also important as these identified possible interactions across (inter-organisational) boundaries. Figure 5 describes the Define Reference Architecture Workflow.

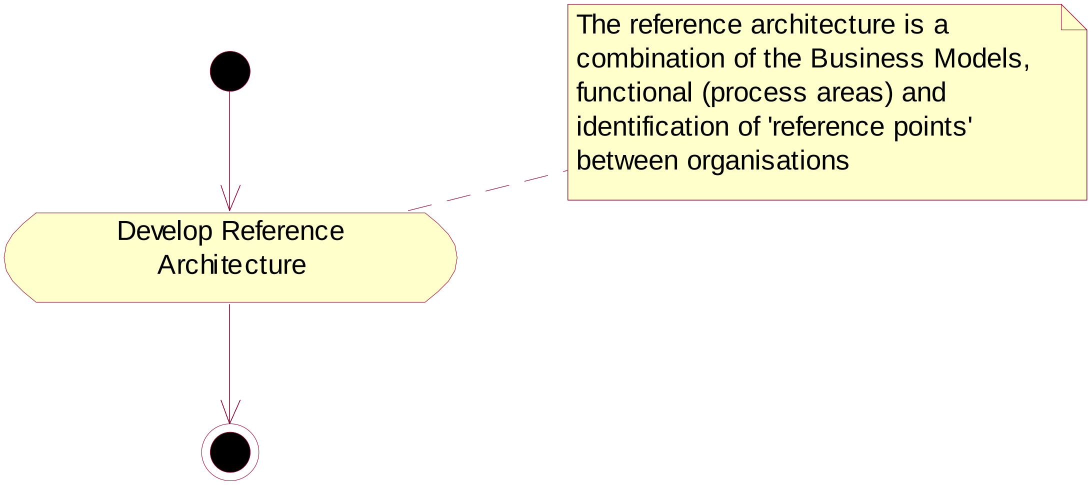

The reference architecture is a combination of the Business Models, functional (process areas) and identification of 'reference points' between organisations
Develop Reference Architecture

Figure 4.5 The Define Reference Architecture Workflow

4.2.4 Example: Reference Architecture for Assurance

In the case study, the reference architecture identifies the process areas and reference points for our business model. Figure 6 presents a snapshot of the Reference Architecture for the IES Management Framework indicating such processes as order handling, SLA negotiation etc. The Reference Architecture can be revisited and refined several times during the guideline execution.

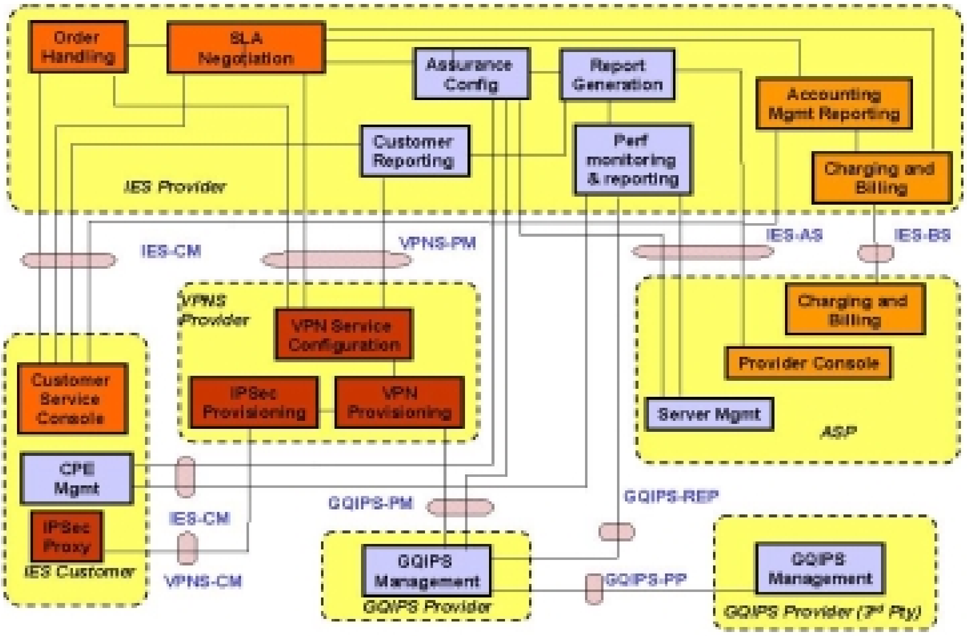


Figure 4.6 Initial FORM Reference Architecture

In the development of the Reference Architecture (for potential Assurance Building Block(s)) it is necessary to identify the key interactions possible between the assurance processes in the different actors (organizations) as well as between other management processes e.g. Billing, Fulfilment. Also interactions which cross organizational boundaries can be grouped together to form Reference Points.

4.2.5 Define Requirements Analysis Workflow

In order to identify candidate functionality and behaviour within the management processes, software requirement specifications and supplementary specifications are developed. Such requirements may be based on a market analysis of customers with regard to the functional areas. Other requirements may be gleaned from standards bodies and published requirement specifications e.g. TM Forum’s requirements for management Building Blocks [GB909]. The use cases and the functionality identified within them, is at the ‘system modelling’ level (rather than the business modelling level). This means is that the requirements modelling work is trying to identify functionality to be supported by computer systems rather the higher-level business activities.

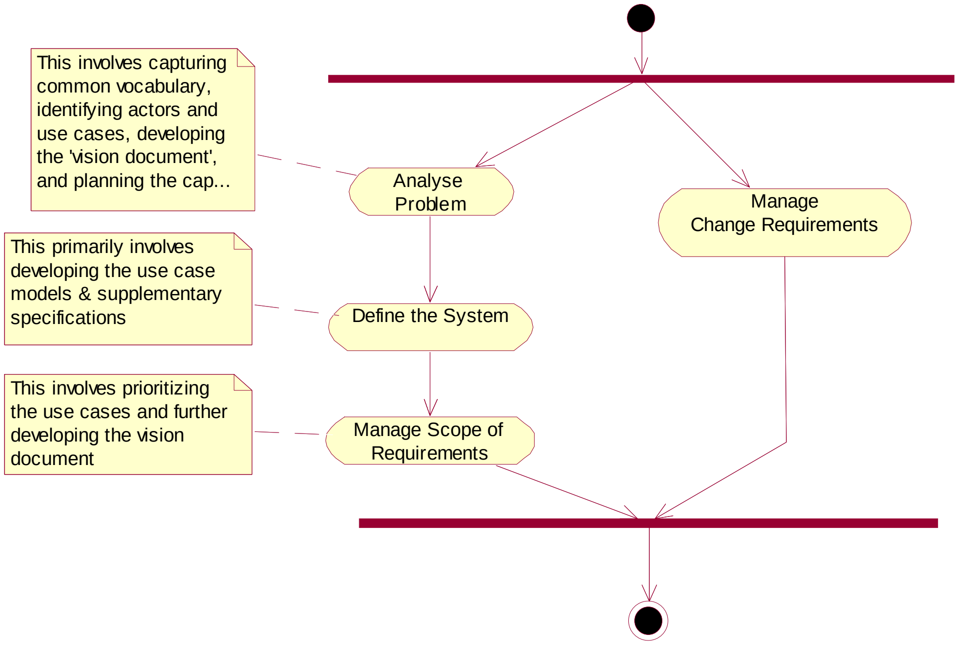

This involves capturing common vocabulary, identifying actors and use cases, developing the 'vision document', and planning the cap...
Analyse Problem
Manage Change Requirements
This primarily involves developing the use case models & supplementary specifications
Define the System
This involves prioritizing the use cases and further developing the vision document
Manage Scope of Requirements

Figure 4.7: Define Requirements Analysis Model Workflow

During these development activities, Use Case Models are developed which describe the desired behaviour of the envisaged systems. Use cases at the boundaries in each of the functional areas are developed. These use cases provide both the actors (roles), which would make use of the management services, and a specification of each of these management services as a use case. The use cases consist of Use Case Model diagrams, supplementary specifications and activity diagrams representing the control flow between the activities. These development activities are customised from the RUP Requirements workflow.

As explained earlier, the end goal of the building block guidelines is to develop building blocks within each functional area, but it is important to develop quite wide-ranging requirement sets and use cases to ensure breadth of coverage for each functional domain.

4.2.6 Example: Requirements Analysis Modelling

Suppose, in the case study, the intended Building Blocks and Building Block Contracts we wish to develop are solely related to Assurance. This workflow helps to define the boundaries and actors, which would be appropriate for an assurance system. Figure 4.8 depicts the use cases actors, and boundaries of an assurance system and Figure 4.9 outlines some of the use cases modelled for the assurance functional domain and Figure 4.10 presents one of the activity diagrams associated with the Assurance Use Cases, namely the activity diagram for the Agree Assurance Support for SLA use case.

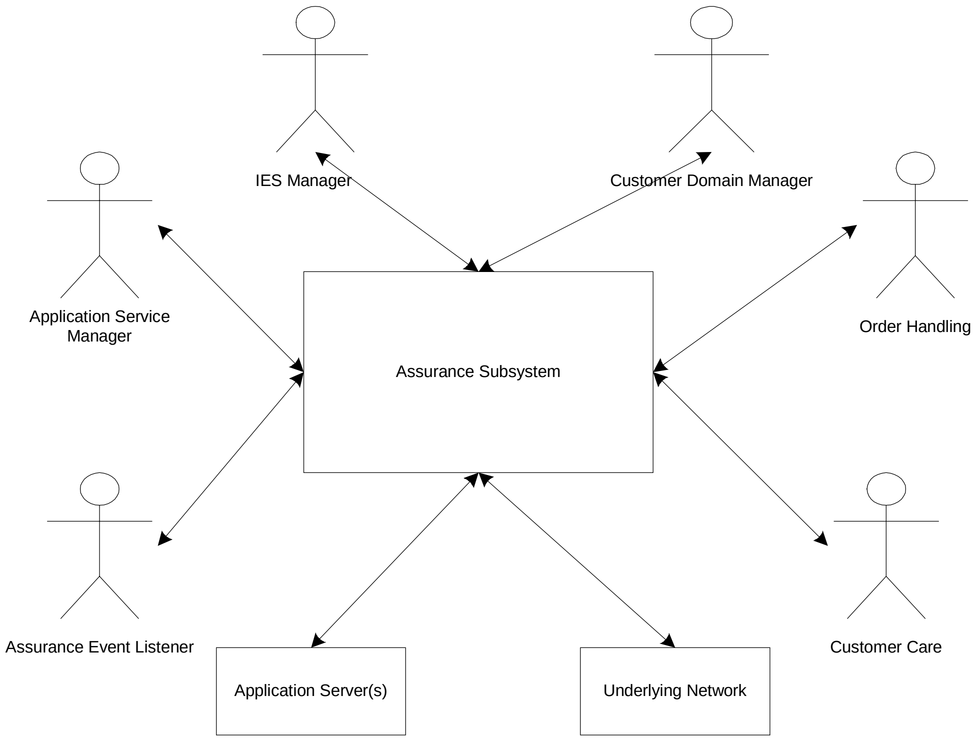


Figure 4.8 Use Case Actors & Boundaries

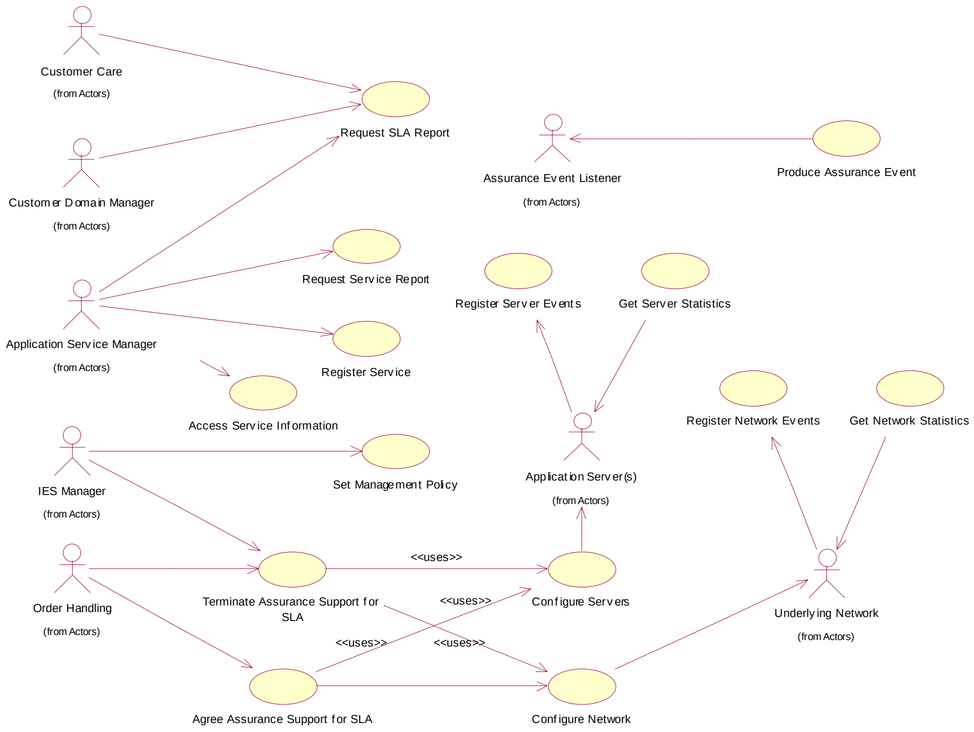


Figure 4.9 Use Case Models (Assurance)

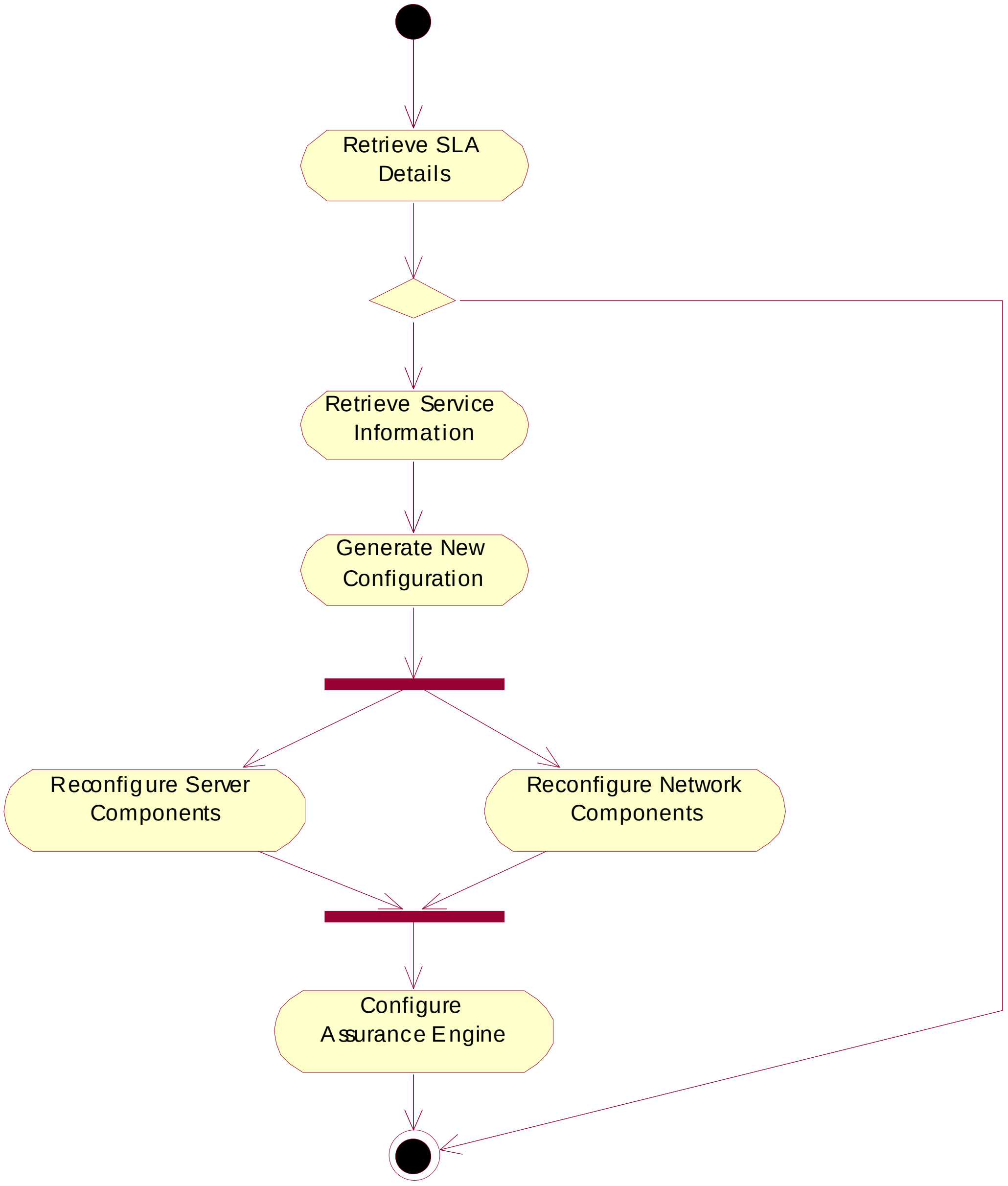

Retrieve SLA Details
Retrieve Service Information
Generate New Configuration
Reconfigure Server Components
Reconfigure Network Components
Configure Assurance Engine

Figure 4.10. Activity Diagram: Agree Assurance Support for SLA

4.2.7 Develop Analysis Models Workflow

This development workflow focuses on the identification of analysis classes and their interactions based on the use cases defined earlier. Artefacts developed during this workflow include Design of Analysis Classes, Collaboration & Sequence Diagrams and

Interfaces. These artefacts can be brought together into subsystems. We can think of a subsystem as a logical collection of classes, which may be useful in forming potential building blocks. Initially these analysis classes can be identified from the Use Cases and activity diagrams developed earlier. In this workflow an initial Information Object Model (captured as a class diagram) is formed.

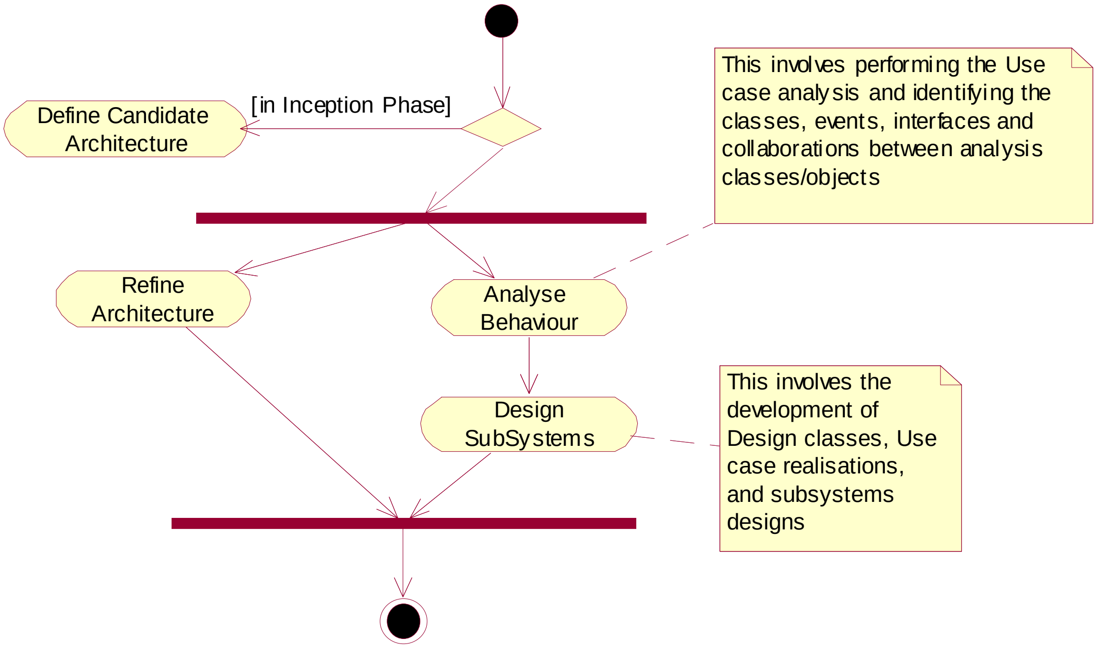

Define Candidate Architecture
[in Inception Phase]
This involves performing the Use case analysis and identifying the classes, events, interfaces and collaborations between analysis classes/objects
Refine Architecture
Analyse Behaviour
Design SubSystems
This involves the development of Design classes, Use case realisations, and subsystems designs

Figure 4.10: Develop Analysis Model(s)

4.2.8 Example: Analysis Object Modelling

In our example case study this workflow defines sets of interacting analysis objects e.g. order handling, config assurance etc as depicted in Figure 4.11. The analysis objects can be sequenced to support the various use cases defined earlier (Figure 4.12)

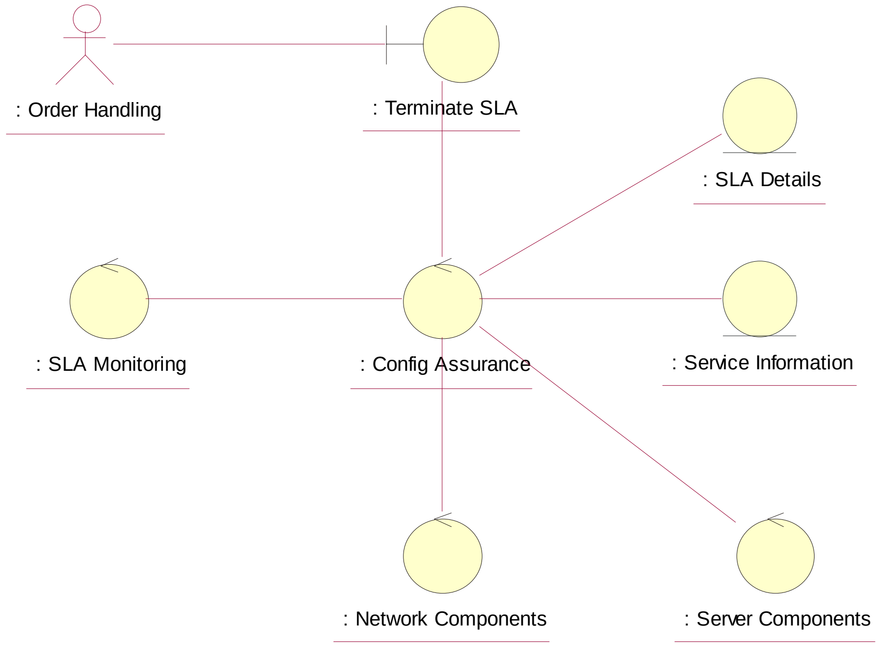


Figure 4.11 Collaboration Diagram: Agree Assurance Support for SLA (Analysis Model)

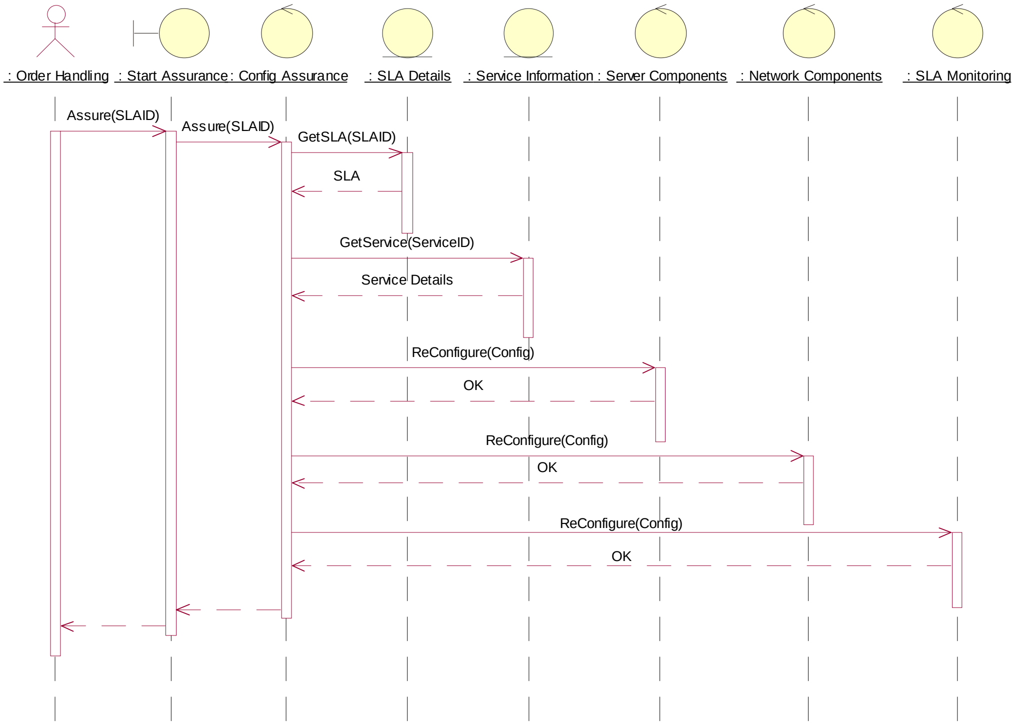


Figure 4.12 Interaction Diagram: Agree Assurance Support for SLA.

4.3 Building Block Development Guideline: Building Block Modelling Phase

This Phase focuses on the revision and refinement of the models and Vision Document(s) completed in the Context Modeling Phase. In this Phase, the Reference Architecture is solidified. Also during this phase, potential Building Blocks and Building Block Contracts are identified and the Use Cases to be supported by the Building Blocks are modelled and refined. The descriptions of Building Block Contracts are further refined and Building Block Contract templates is populated. A Building Block Contract template defines the essential artifacts, including UML models, required to characterize and describe the Building Blocks Contracts.

The main artifacts of the Building Block Modeling Phase are:

- A stable system architecture model.
- Development models for Building Blocks Contracts in the FORM Framework.
- Development of models for Building Blocks in the FORM Framework

4.3.1 Process workflows in the Building Block Modelling Phase

The workflows in the Building Block Modeling Phase include those of the earlier phase, as the overall development process is iterative. However, the workflows Perform Business Modeling, Define Reference Architecture, Perform Requirements Analysis, and Develop Analysis Object Models are re-iterated so as to refine the relevant artifacts. An extra Workflow in this phase is the Re-organisation of Analysis Objects into Building Blocks.

Model (Re)Organisation Workflow: This workflow identifies the development activities involved in Re-organising the Analysis Classes/Models in order to group useful behaviours/entities into potential Building Blocks. One reason for this re-organisation may include the decision to adopt/use standard information models or to suggest interfaces and information models (e.g. IPDR for Accounting, CIM for Assurance). Another reason for re-organisation is that the analysis work up to this point, has mainly been performed to identify candidate system-wide functionality & information entities, and the development activities now focuses more explicitly on Building Blocks and Building Block Contracts. There are many possible criteria, which are useful in identifying potential Building Blocks (or components).

4.3.2 Re-organise Analysis Model(s) Workflow

This workflow is concerned with the identification of candidate building blocks from amongst the analysis objects identified in the earlier development activities. Essentially a Building Block is an atomic unit of software deployment and management. Building Blocks can support multiple interface types and multiple instances of those types. A Building Block Contract specifies a grouping of information and behaviours, which can be re-used to support management business processes. One or more Building Block Contracts can be supported (implemented) by a Building Block in the FORM Framework.

The notion of a Building Block and Building Block Contract in the Open Development Framework differs from the notion of ‘component’ in some development processes (e.g. RUP), in that the Building Block can support one or more contracts and is described as a package of modeling artifacts. For example, in RUP, the term component only has meaning in the deployment model and not in the analysis/development activities. Thus the closest RUP notion to Building Block is that of ‘subsystem’ which can be a collection of related development artifacts representing some functionality. However, the Open Development Framework’s definition of a Building Block is much more specific and prescriptive than that of a ‘subsystem’ in RUP.

A Building Block description includes use cases and collaboration diagrams to indicate usage scope of the Building Block, a contract interface specification and a specification of information classes passed into or out of the building blocks. A Building Block supports one or more contact specifications.

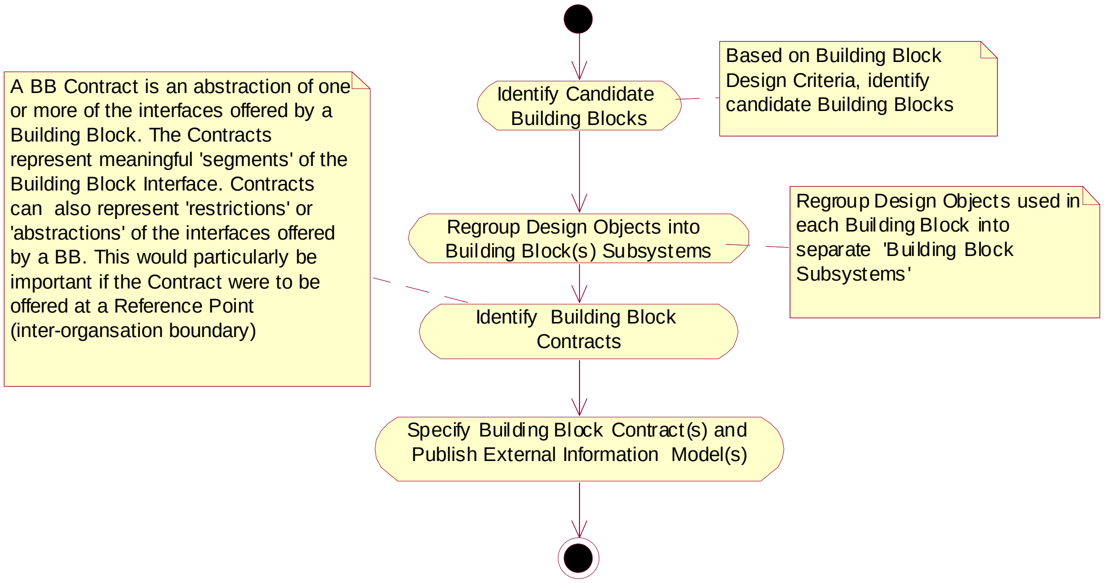

A BB Contract is an abstraction of one or more of the interfaces offered by a Building Block. The Contracts represent meaningful 'segments' of the Building Block Interface. Contracts can also represent 'restrictions' or 'abstractions' of the interfaces offered by a BB. This would particularly be important if the Contract were to be offered at a Reference Point (inter-organsation boundary)
Identify Candidate Building Blocks
Based on Building Block Design Criteria, identify candidate Building Blocks
Regroup Design Objects into Building Block(s) Subsystems
Regroup Design Objects used in each Building Block into separate 'Building Block Subsystems'
Identify Building Block Contracts
Specify Building Block Contract(s) and Publish External Information Model(s)

Figure 4.12 Reorganise Analysis Model Workflow

4.3.3 Example: Reorganising Analysis Models for modeling Building Blocks

In the case study the analysis objects are regrouped and remodeled into packages (represented diagrammatically as folders). The result of this reorganisation of classes in to logically independent packages identifies candidate Building Blocks.

This reorganisation can be based on shared information requirements, shared objectives, or the need for close collaboration. However, it is important to note that the classes within a single package need not be all of the same type i.e. some may provide persistency, others represent business logic or control objects. Figure 4.13 identifies several packages (candidate Building Blocks) each containing objects.

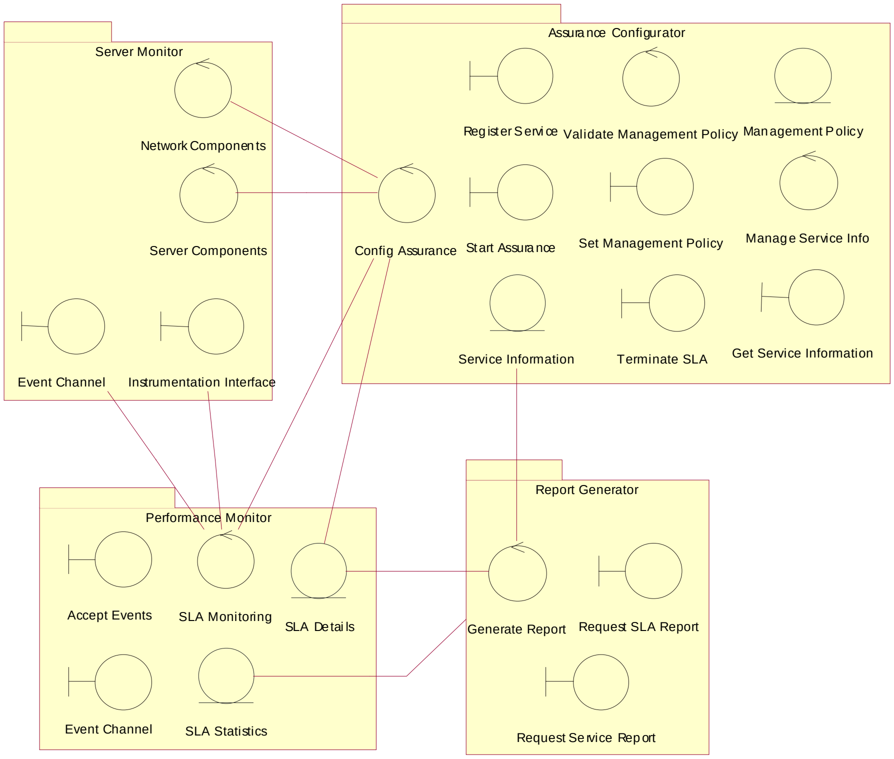


Figure 4.13 Grouping of Analysis Object into Building Blocks

Sequence diagrams can also be modelled to show how the candidate building Blocks can be sequenced to support use cases identified earlier in the guideline.

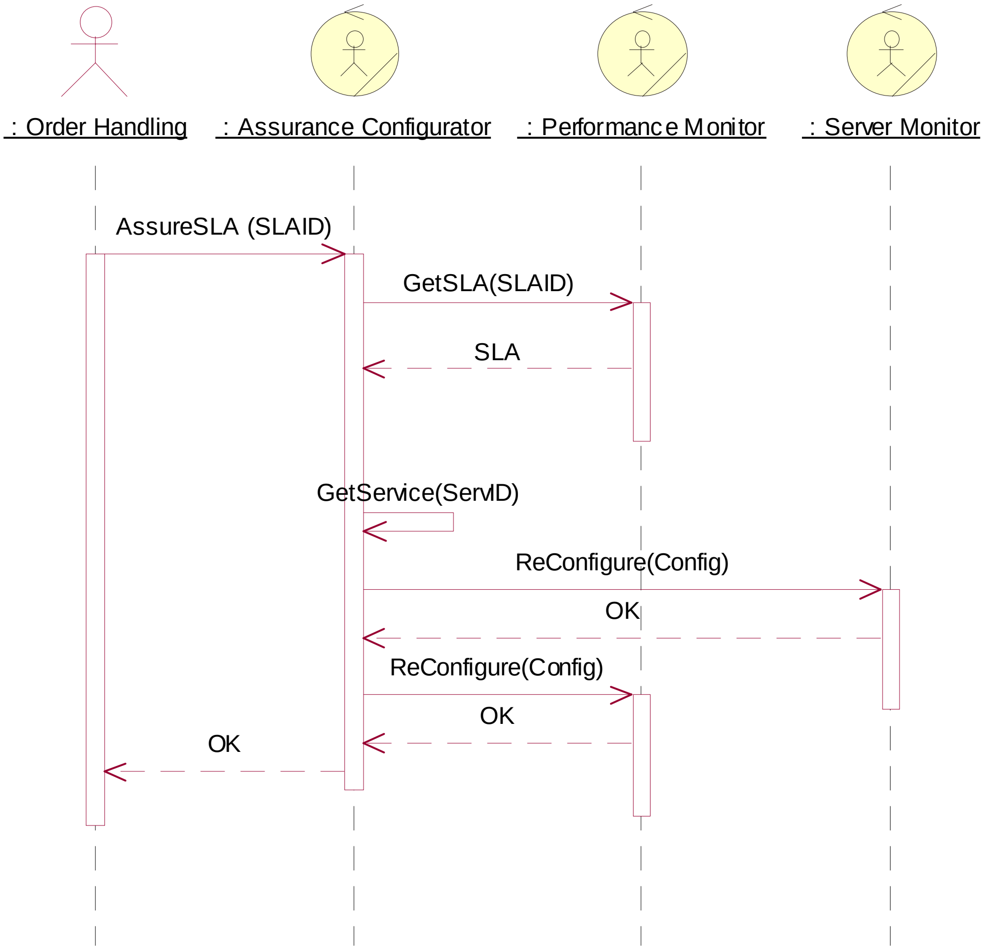


Figure 4.14. Interaction Diagram: Agree Assurance Support for SLA

4.3.4 Modelling Candidate Building Block Contracts

The Open Development Framework’s template for describing a Building Block Contract is as follows:

- A Building Block Contract Name (specified as text).
- The names of defined Reference Points Supported by the Building Block Contract (if any) and the Business Role supporting the services provided by the Contract. These are points on the abstract organisational boundaries through which the Building Block Contract may be accessed.
- Contract description defining service offered by Building Block Contact (specified as text).
- Use cases & collaboration diagram(s) to illustrate usage scope of Building Block Contract

- Contract interface specification.
- Information Objects communicated at the interface of the Building Block Contract i.e. UML Class Diagram of information objects exchanged by the contract. This is called the Boundary Information Model for the Building Block Contract
- Technological description for Building Block Contract (specified as text).
- Collaboration diagrams illustrating the Building Block Contract potential interactions with other Building Block Contracts in the FORM Framework. The inclusion of these collaboration diagrams is optional as they are intended only to indicate where close reliance or relationships exist between Building Block Contracts.

A non-exclusive set of criteria is defined to assist in identifying candidate Building Blocks by reorganising the analysis classes identified in the previous development activity:

- Does the grouping of classes provide an Enterprise Wide information service, re-usable business logic or generally useful User Interface (i.e. at the Enterprise Information Tier, Process Automation Tier and Human Interaction Tier)?
- Does the grouping of classes represent some self-contained behaviour (logical grouping of closely related behaviours)?
- Is the level of inter-dependence between a set of classes (collaborating classes based on original use cases) sufficient as to suggest their close dependence?
- Is there a definite ‘service’ or ‘services’ that a group of classes can uniquely support (does it add a useful, distinct, service to the system)?

There are two possible approaches to the specification of contracts. The first is to design the Building Block and then design the contract specifications, which that building block can support as a set of abstractions on the building block interface (as indicated in the activity diagram earlier). It is important to note that the contracts can offer different functions/interfaces or can support restrictions (or abstractions) on the interfaces supported by the building blocks. An alternative approach to defining Building Block Contracts is to attempt to design the contracts first an then define the building blocks to support such contracts.

4.3.5 Example: Modelling Building Block Contract

A full example Building Block Contract specification and description is presented separately in section 5.

4.4 Building Block Development Guideline: Building Block Implementation

The Building Block Development Guideline is focused on prescribing the workflows and modeling artifacts needed for BB Contract and BB development. The actual implementation and coding of the building blocks are outside the scope of the guideline. However, many commercial UML based development tools provide code generation facilities. These facilities can generate skeletal code in a variety of programming languages and middleware technologies. The use of such tools and the resultant programming are not part of the guideline.

5. Case Study: Example Building Block Specification

In order achieve a better understanding of Building Blocks and their Contracts, this section presents the specification of a building block, which was developed for the FORM Assurance Domain. The case study illustrates a Building Block Contract specification and the UML models used to describe them. It is important to note that although, in this specification, only a single contract interface is specified, multiple interfaces are also permissible for a Building Block Contract.

5.1 Specification of QoS Server Monitor Building Block Contract

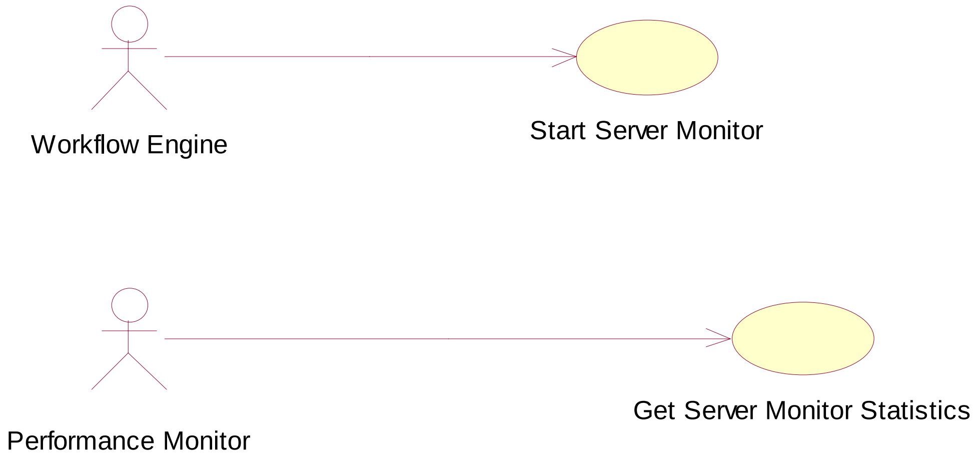
BB Contract Name:	Server Monitor
Reference Points:	IES-SM

Contract Description:

This contract allows access to the CIM information base stored in the Server Monitor building block. The building block monitors server statistics, calculating secondary combinatory statistics when necessary. Both primary and secondary statistics are stored within the information base for retrieval. Objects facilitating the management of the Server Monitor itself are also present in the information base. These objects perform a number of different tasks such as initialising and managing downloadable extensions to the module.

5.1.1 Example: QoS Server Monitor Use Cases

The use cases should define the services offered at the systems boundary as well as the actors who would use the management service. The figure below identifies two assurance management services, namely start server monitor, and get server monitor statistics. These management services are depicted being used by the workflow engine (in the service assurance trial system, the various assurance business processes are initiated and integrated by a workflow engine) and a performance manager (another building block in the system).



5.1.2 Example: Building Block Contract Interface

The contract consists of the interface shown in figure 3.2 below.
The cimOperation parameter should contain a complete XML document that conforms to the "CIM Operations Over HTTP" standard specified by the DMTF. This standard specifies the structure of an XML document used to query and otherwise manipulate the CIM information base. It also specifies the format of the XML response to these requests (i.e. the returned string). The final item to note is that this method can also throw an exception to indicate that the cimOperation did not conform to the CIM DTD and therefore could not be processed.

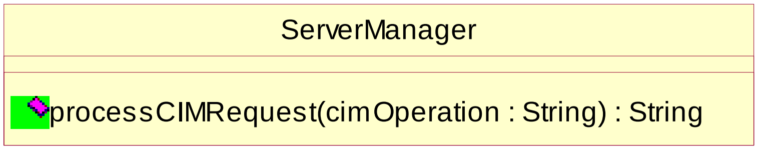


Figure 3.2 - UML Diagram of BB Contract Interface

5.1.3 (Building Block) Boundary Information Model

The external visible information model (termed the Boundary Information Model) supported by the building block contract is in the form of CIM classes and objects. It is therefore important to understand the structure of these classes and how they relate to each other. The Boundary Information Model for this Building Block Contract can be logically divided into four aspects the Server Monitor Management Information Model, Server Monitor Configuration Information Model, Service Description Information Model and the Calculated Statistics Information Model

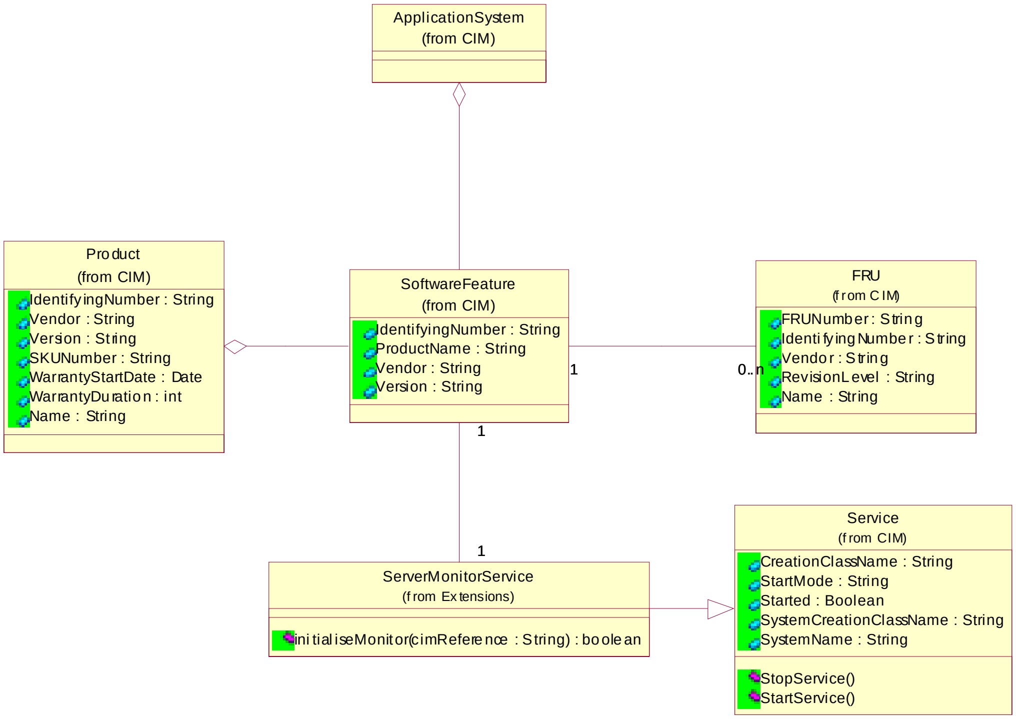


Figure 3.2 Server Monitor Information Model

Associated with each Building Block Shared Information Model is an explanation of the classes in the building block.

Class	*Usage*
CIM_ApplicationSystem	***Maintains high level information about the Server Monitor such as contact information etc.***
CIM_SoftwareFeature	***Describes the features of Server Monitor. There must be one instance called “Statistical Calculation”. Other instances may be made as deemed necessary.***
CIM_Product	***An instance of this class may or may not be provided. If provided it’s only purpose is to represent how certain software feature comprising a product.***
CIM_FRU	***Associated with the “Statistical Calculation” instance will be zero or more instances of CIM_FRU. Each of these instances will represent a piece of code downloaded to the component to aid in the calculation of statistics.***
FORM_ServerMonitorService	***One instance of this class must be associated with the “Statistical Calculation” instance. This class provides the method by which the Server Monitor is initialised and eventually stopped.***

5.1.4 Collaboration Diagram of Server Monitor BB with other FORM Framework BBs

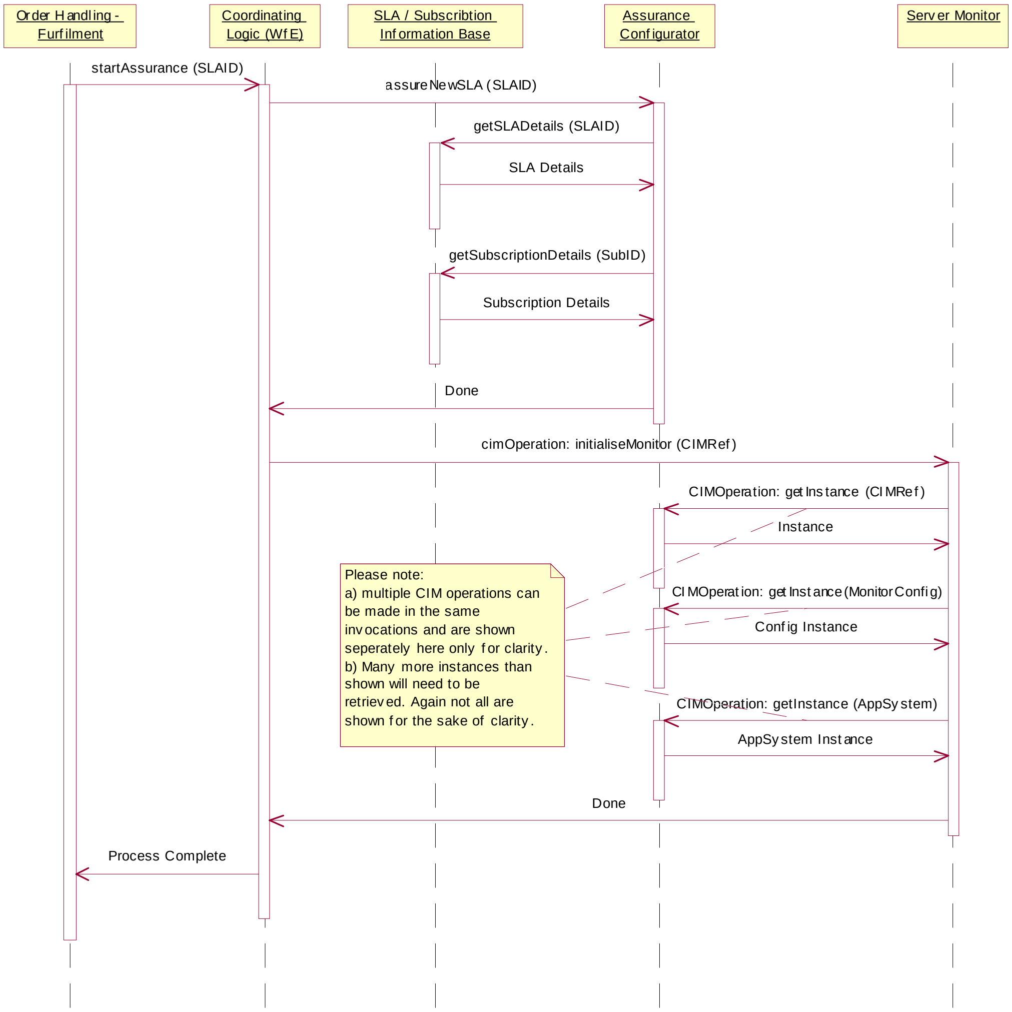

Order Handling - Furfilment
Coordinating Logic (WfE)
SLA / Subscribtion Information Base
Assurance Configurator
Server Monitor
startAssurance (SLAID)
assureNewSLA (SLAID)
getSLADetails (SLAID)
SLA Details
getSubscriptionDetails (SubID)
Subscription Details
Done
cimOperation: initialiseMonitor (CIMRef)
CIMOperation: getInstance (CIMRef)
Instance
CIMOperation: getInstance(MonitorConfig)
Config Instance
CIMOperation: getInstance (AppSystem)
AppSystem Instance
Done
Process Complete
Please note:
a) multiple CIM operations can be made in the same invocations and are shown seperately here only for clarity.
b) Many more instances than shown will need to be retrieved. Again not all are shown for the sake of clarity.

Figure 3.3 Server Monitor Collaboration diagram

The following is the XML specification of a Contract specification from the Server Monitor BB:

- -
<BBContract name="**cs.tcd.ie/FORM/ServerMonitor**" contractSpecifier="**Brian Cullen**" date="**8/8/2001**" version="**1.0**" xmlns="**http://www.ist-form.org/BBContractDescription**" xmlns:xsi="**http://www.w3.org/2001/XMLSchema-instance**" xsi:schemaLocation="**http://www.ist-form.org/BBContractDescription C:\College\FORM\BB-Contract-Description-v2-1.xsd**">
<description>**This contract allows access to the CIM information base stored in the Server Monitor building block. This building block monitors server statistics, calculating secondary**

```
    combinatory statistics when necessary. Both primary and
    secondary statistics are stored within the information base for
    retrieval. Objects facilitating the management of the Server
    Monitor itself are also present in the information base. These
    objects perform a number of different tasks such as
    initialising and managing downloadable extensions to the
    module.</description>
-                                              <supportedReferencePoints
    referenceArchitectureURI="http://www.ist-
    form.org/ReferenceArchitecture">
    <referencePoint name="IES-CM" />
    <referencePoint name="GQIPS-PM" />
    <referencePoint name="IES-AS" />
  </supportedReferencePoints>
- <contractScope>
  - <useCaseSet diagramURI="UseCase01.html">
                                              <useCaseCollaboration
        diagramURI="UseCaseCollaboration01.html"
        relatedUseCase="Configure Monitor" />
                                              <useCaseCollaboration
        diagramURI="UseCaseCollaboration02.html"
        relatedUseCase="Get Statistics" />
                                              <useCaseCollaboration
        diagramURI="UseCaseCollaboration03.html"
        relatedUseCase="View Configuration" />
    </useCaseSet>
  </contractScope>
- <interfaceInteractions>
     <interfaceInformation description="CIM Operations Format"
      fileURI="InterfaceInformation01.html" />
      <interfaceInformation description="EJB CIM Interface"
      fileURI="InterfaceInformation02.html" />
     <interfaceInformation description="CIM Service Description"
      fileURI="InterfaceInformation03.html" />
      <interfaceInformation description="CIM Server Monitor
      Settings" fileURI="InterfaceInformation04.html" />
      <interfaceInformation description="CIM Server Monitor"
      fileURI="InterfaceInformation05.html" />
     <interfaceInformation description="CIM Calculated Statistics"
      fileURI="InterfaceInformation06.html" />
  </interfaceInteractions>
- <boundaryInformationModel>
    <informationModel modelURI="BoundaryInfoModel01.html" />
    <informationModel modelURI="BoundaryInfoModel02.html" />
    <informationModel modelURI="BoundaryInfoModel03.html" />
    <informationModel modelURI="BoundaryInfoModel04.html" />
  </boundaryInformationModel>
   <technologyDescription>All information accessed and passed
    through this contract is done so in CIM format.In particular
    this contract supports the "CIM Operations over HTTP"
    standard specified by the DMTF. This standard specifies the
```

```
        structure of an XML document used to query and otherwise
        manipulate the CIM information base. It also specifies the
        format of the XML response to these requests (i.e. the
        returned string).</technologyDescription>
</BBContract>
```

6. Business Process Driven System Development Guideline

As stated previously the goal of the Business Process System Development guideline is to facilitate management systems integrators to construct management solutions from Building Block Contracts. The Guidelines provides employs a 'Business Process Driven' approach to management system construction from re-usable Building Blocks by explicitly modeling the required system processes and their constituent system activities. The guideline uses these system activities to determine the Building Block Contracts needed to implement these processes. Typically this guideline is used by system integrators and service operators where they need to implement managed solutions using existing Building Blocks and Building Block Contracts offered in the ODF[3].

The Guideline itself is divided into eight process workflows. Each workflow has a specific objective and produces or refines model(s) or artifacts. The workflows iterate the classic development activities from business modeling, requirements capture & management, system analysis and design modeling, implementation and testing [Fowler 97].

The guideline specifies mapping the system activities to Building Block Contracts to encourage the reuse of existing Building Block Contracts in implementing the management system processes. This mapping is at the heart of the reuse of Building Block Contracts in the implementation of management processes. A second part of the mapping of management activities to Building Block Contracts is the reconciliation of External Information Model(s) of the Building Blocks Contracts to the information flows in the system processes.

Figure 1 identifies the principal workflows involved in the Guideline. These involve:

(i) *Performing Business Modeling:* This workflow facilitates the definition of Business Roles, business use cases and organisational units. The key results of this workflow are the development of Business Use Case Model(s), Business Model (representing the business roles and organizational units), and a refinement of the Reference Model for the ODF framework (i.e. a specialization of the ODF reference model indicating the management processes and reference points to be used).

(ii) *Define Requirement Analysis:* This workflow facilitates the identification of candidate behaviour of the management (business) processes, software requirement specifications and supplementary specification. The key result of this workflow is the system use cases and supplementary use case specifications

[3] It is worth noting that the early workflows of the Business Process Driven System Development Guideline are similar to those of the Context Modeling Phase of the Building Block Development Guideline. This is entirely appropriate as both initially map out the business models and requirements context of the systems under construction. However the BB Development Guideline is used for developing BBs for the ODF where as the BP System Development Guideline is intended for users of the ODF to develop management solutions.

(iii) *Perform System Process and System Information* Modeling: In this workflow the required system process(s) are represented as system activity diagrams. Thus this workflow facilitates the modeling of system activities, their control flow, and their information flows. The key results of this workflow are system activity diagram(s) representing the system processes to be implemented.

(iv) *Re-Model System Processes and Map to Building Block Contracts:* This workflow allows the mapping of system activities (and information flows) to Building Block contracts. This is one of the most important workflows in the guideline. In this workflow the system activities are decomposed or aggregated to match, as closely as possible, available Building Block Contract interface specification. This involves matching the BB Contract interface function(s) as well as their information requirements. Where matching is possible, the system activities are annotated with the Building Block Contract which support it. Where the matched Building Block Contract requires extra information, these extra information objects have to be included in the system process. Where matching is not possible, the system activities will be modeled as bespoke system objects which require separate design and implementation.

The key result of this workflow is the system process modeled as activity diagrams, with (some of the) system activities annotated with the Building Block Contract associated with them. The information objects in the activities diagrams are a combination of information objects drawn from the Building Block Boundary Information models and Information Objects developed specially for the process (i.e. bespoke information objects).

(v) *Model Missing Objects and Information Workflows:* This workflow supports the modeling of system objects and information which is not supported by the chosen Building Block Contracts. This workflow concentrates on the bespoke development of management system functionality/objects which have to be developed, as there is no appropriate Building Block Contracts to readily support it. The system objects are modeled as use cases, activity diagrams and class diagrams. Where a system activity involves the design of several system objects, they are grouped together in a subsystem package.

(vi) *Implement Building Block Integration:* This workflow facilitates the integration of the Building Blocks so that the intended system processes are executable. This step may be automated if the design tool (used for modeling the system activity diagrams) is capable of generating an XMI description of the system activity model. Such an XMI description can be used to determine the control flow, data flow and appropriate Building Block Contract invocations. Depending on the integration implementation approach, such an XMI description could be used to populate a Business (Logic) Object capable of making the necessary decisions and invocations on the appropriate Building Contracts (interfaces), or could be used to populate a workflow specification for a workflow execution environment. The key result of this workflow is the implementation of the control and information flow for the business process.

(vii) *Map Building Block Contracts to Building Blocks and deploy BBs:* This workflow facilitates the selection of Building Blocks to be deployed in the system. This workflow can also involve identification and placement of technology and data gateways where building blocks are implemented using different technologies (be it Business (Logic) object or workflow integration engine). This would be needed if the Building Block Contracts were technology neutral (i.e. the interface description of the Building Block Contract were not described in a particular distributed implementation technology). In this way the technological or information representation heterogeneity of the Building Blocks can be hidden from the integration business logic. Where the Building Block Contracts are specified using technology specific interfaces, there is no need for these gateways as the Business (Logic) Object or workflow engine would be generated with the required technology specific invocations.

(viii) *Perform testing and Deployment:* This workflow defines and executes the testing necessary for the management process execution. This involves generating test plans and execution of those plans.

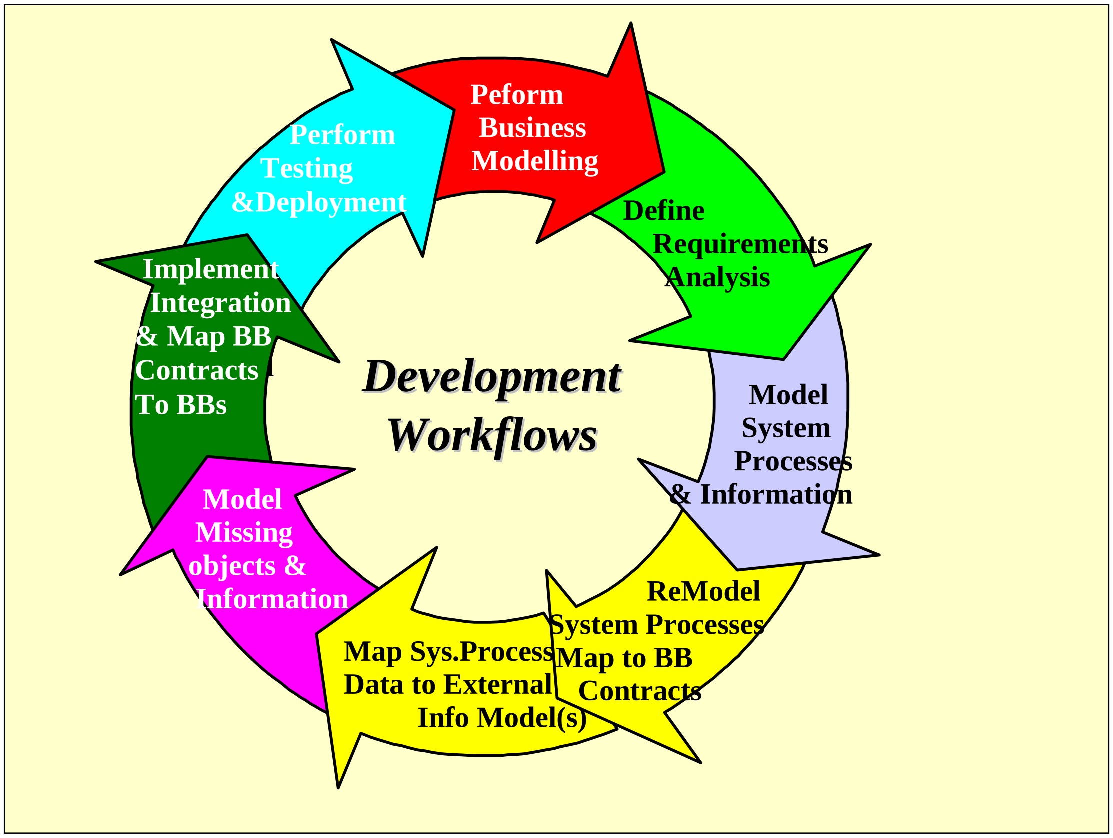

Peform Business Modelling
Define Requirements Analysis
Model System Processes & Information
ReModel System Processes Map to BB Contracts
Map Sys.Process Data to External Info Model(s)
Model Missing objects & Information
Implement Integration & Map BB Contracts To BBs
Perform Testing &Deployment
Development Workflows

Figure 6.1 Overview of Process Driven System Development Guideline

6.1 Perform Business Modelling and Reference Architecture Refinement Workflow

This process workflow facilitates the definition of business model(s) based upon management business processes. This involves identifying Business Roles, Business information entities, Business Use Cases and Organisation Units.

This workflow seeks to define the Business Roles, Business Use Cases, Business Processes and Business resources/information entities with which the Actors interact. There are two aspects from which the Business Modelling is performed. The first is the External View of the Business, the second being the internal view of the system. To model Management (Business) Processes the following models/diagrams are specified

(iv) Business Use Case Diagram(s): depicting business roles (workers and/or organisations), use case name (external view)

(v) Use Case Realisation which models the business workers and entities/resources needed to carryout the use case (internal view)

(vi) Activity Diagram depicting the activities involved in carrying out the use case (internal view).

Therefore the workflow can be summarised as Figure 6.2

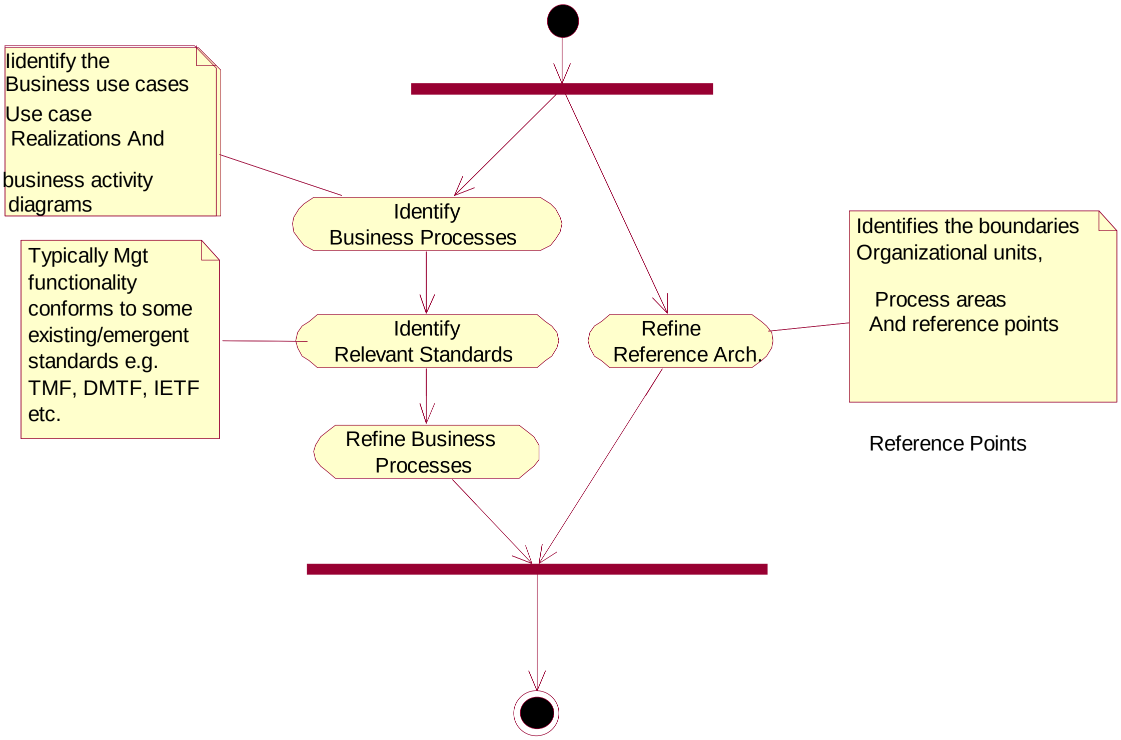


Figure 6.2 Perform Business Modeling and Reference Architecture Refinement Workflow

In developing the Business Model, a customisation of the Open Development Framework's Reference Architecture can be made. The Reference Architecture identifies reference points between organisational boundaries, the placement of desired management processes within these boundaries and the relationships (potential interaction) between these process areas across organisational boundaries. The Business Modeling workflow therefore may involve either extending or simplifying the reference architecture to suit its business circumstances and context of the management processes to be developed.

Several external influences will very probably influence both the refinements of the reference model and the business model produced. These include standardisation in the management process and technology areas e.g. TeleManagement Forum, DMTF as well as standards specific to the management process domain e.g. IETF/DMTF Quality of Service standards, IPDR accounting standards. It is important to identify the relevant standardisation influences as this can have specific effects on system activities, use cases and information models.

Key artefacts produced by this process workflow include:

- A textual description of the scope of the envisaged management processes and the business organisation and roles involved in these processes.
- Business Use Case Model(s),
- Business Model (representing the business roles and organizational units), and
- A refinement of the Reference Model for the ODF framework (i.e. a specialization of the ODF reference model indicating the management processes and reference points to be used).

6.1.1 EXAMPLE: Assuring an Educational (WWW) based service

Suppose a Customer Organisation wishes to subscribe to a Web based Application Service (offered by an Application Service Provider). The Customer Organisation may wish some guarantees regarding the quality of service for the web based services and the network performance upon which these services are offered. Such Quality of Service management is offered by an Inter Enterprise Service Provider (IESP). The network quality of service could be offered by a Virtual Private Network Service (VPNS) Provider. This IESP establishes relationships with the ASP and the Customer and Internet Service Providers that link the ASP and Customer. The IESP agrees a guaranteed quality of the VPN with the VPNS provider. It may be that this VPNS provider, in turn utilizes a Guaranteed Quality IP provider to ensure the quality of service over the IP connections between the end customer and the application service provider.

Typically the business modeling work would first identify the important business use case.

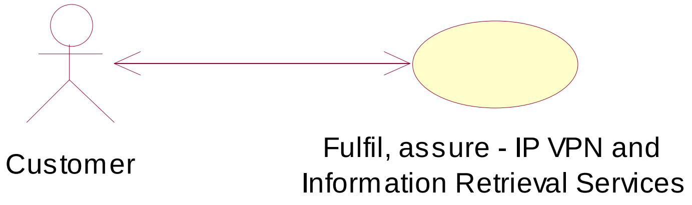


Figure 6.4: Fulful & assure the Educational Information Service and VPN.

The Business Model would therefore look like:

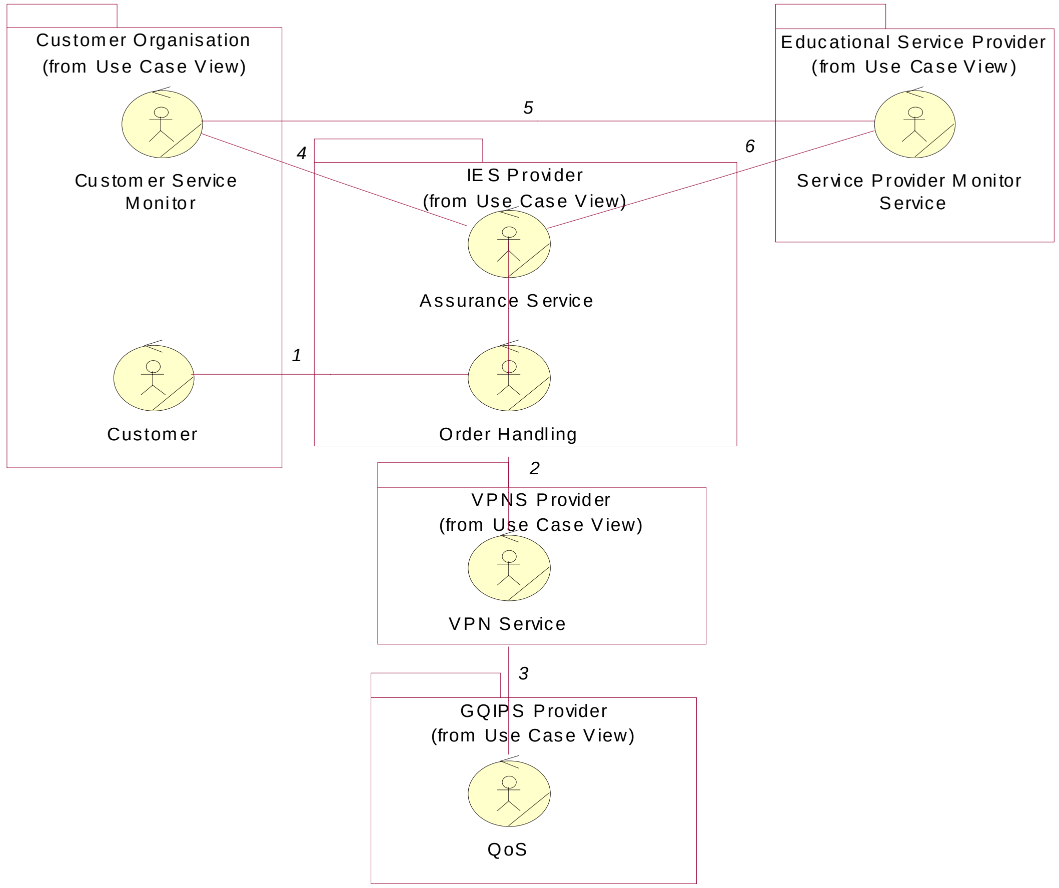


Figure 6.5 Business Model for Assurance Management Processes

The model depicts the organizations using the folder representation, the business roles as business workers and the relationships between the organizations. Also modeled in this workflow is the customized version of the Reference Architecture as depicted in Figure 6. This customized reference architecture identifies the process areas within the organizations and the inter-organisational reference points operating between these organizations.

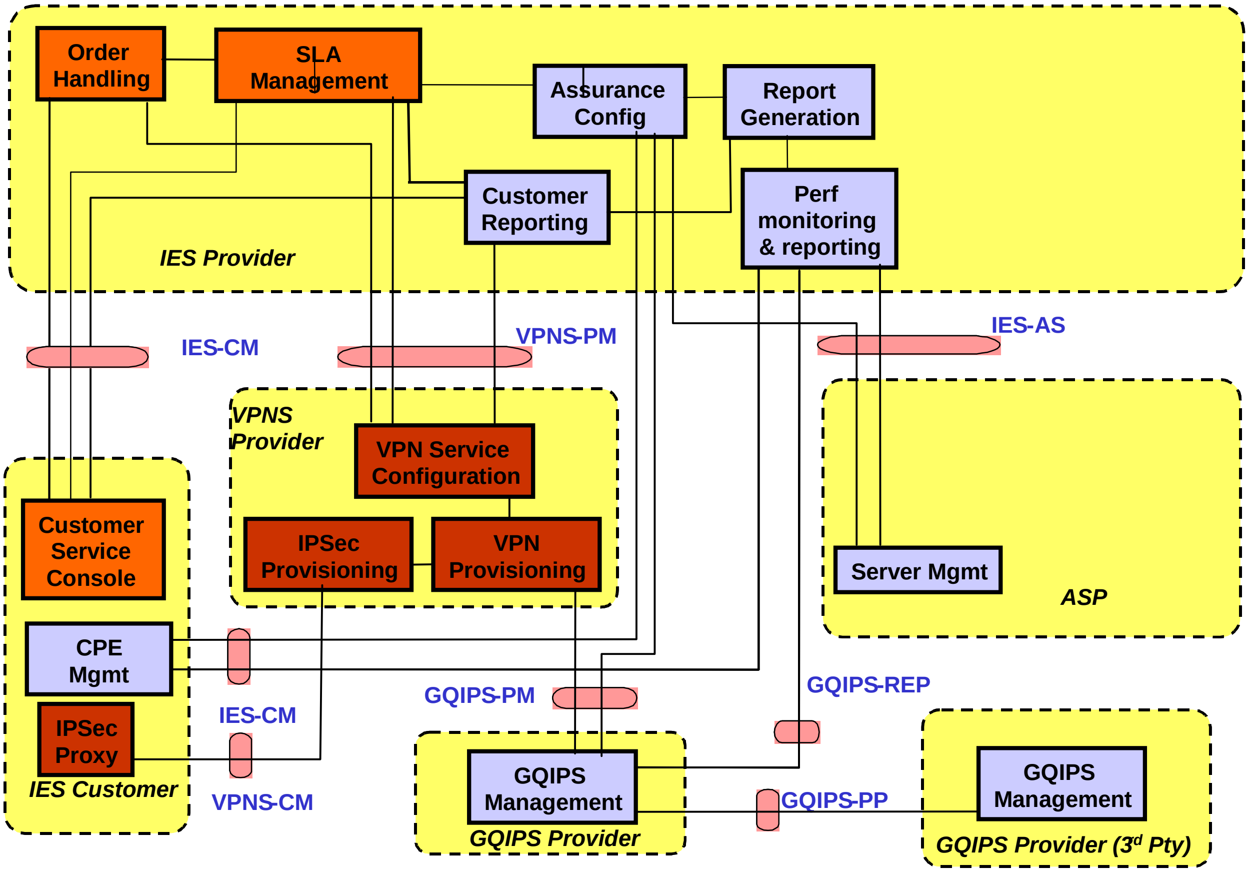


Figure 6.6 Reference Architecture for Fulfilment & Assurance Processes.

This reference architecture identifies the names of the processes relevant to the business use cases, the inter and intra process interactions within and between the organizations identified in the business model and the reference points through which inter-organisational processes interact.

6.2 Define Requirements Analysis Workflow

In order to identify candidate behaviour of the management processes, software requirement specifications and supplementary specifications are developed. Such requirements may be based on a market analysis of customers with regard to the functional areas. Other requirements may be gleaned from standards bodies and published requirement specifications. The use cases and the functionality identified within them, is at the 'system modelling' level (rather than the business modelling level). What this means is that the requirements modelling work is trying to identify, functionality to be supported by computer systems rather than higher-level business activities.

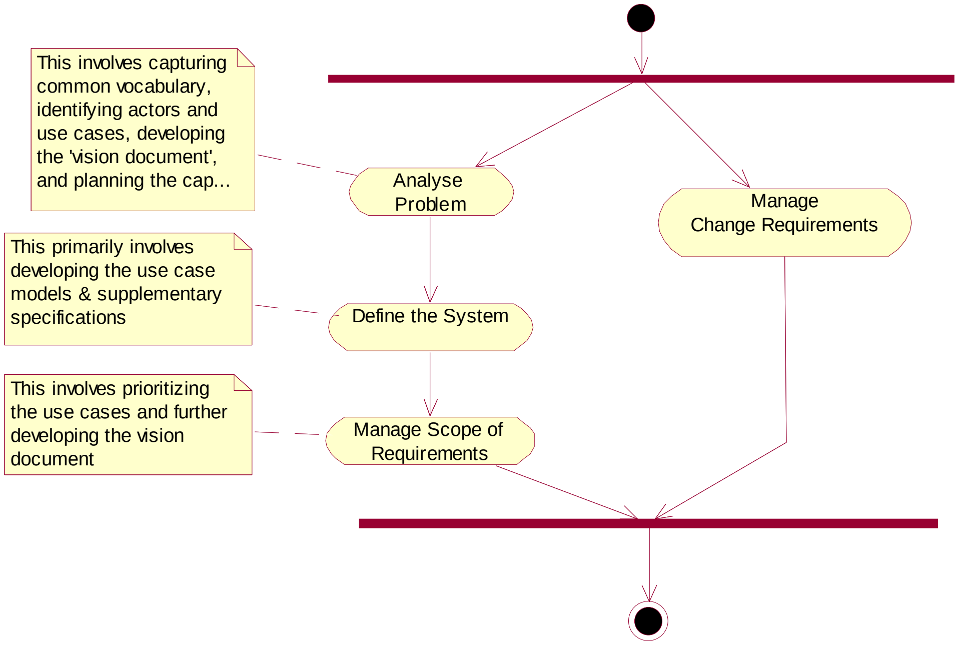

This involves capturing common vocabulary, identifying actors and use cases, developing the 'vision document', and planning the cap...
Analyse Problem
Manage Change Requirements
This primarily involves developing the use case models & supplementary specifications
Define the System
This involves prioritizing the use cases and further developing the vision document
Manage Scope of Requirements

Figure 6.7: Define Requirements Analysis Model Workflow

During these development activities, Use Case Models are developed which describe the desired (SYSTEM) behaviour of the envisaged management systems. Use cases at the boundaries in each of the process areas are developed. These use cases identify both the actors (roles), which would make use of the management services, and a specification of each of these management services as a use case. The use cases consist of Use Case Model diagrams, supplementary specifications and activity diagrams representing the control flow between the activities.

Thus the key artifacts produced by this workflow are:

- System use cases and supplementary use case specifications

6.2.1 Example Use Case Model for Fulfilment-Assurance

Taking the same example as before, the use case model for the IES provider could involve the initialization of the Service Level Agreement (SLA) between the end customer and the IES provider, the conclusion of the SLA negotiation and, as a consequence of this SLA negotiation conclusion, the instigation of the configuration of the assurance service, VPN and GQIPS. The folder notation in Figure 6.8 indicates that the use cases are contained with various roles defined earlier in the business model.

Although not shown here, these use cases diagrams would also have use case requirements specifications developed for them.

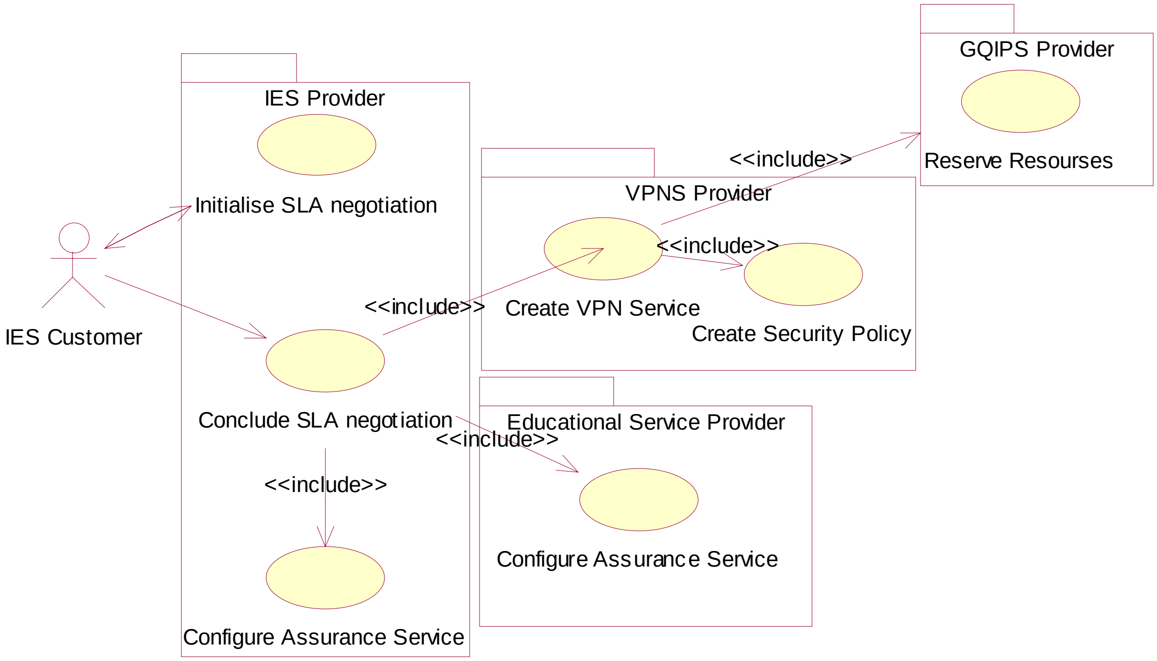


Figure 6.8: Example Use Model for assurance configuration

6.3 Perform System Process & System Information Modelling Workflow

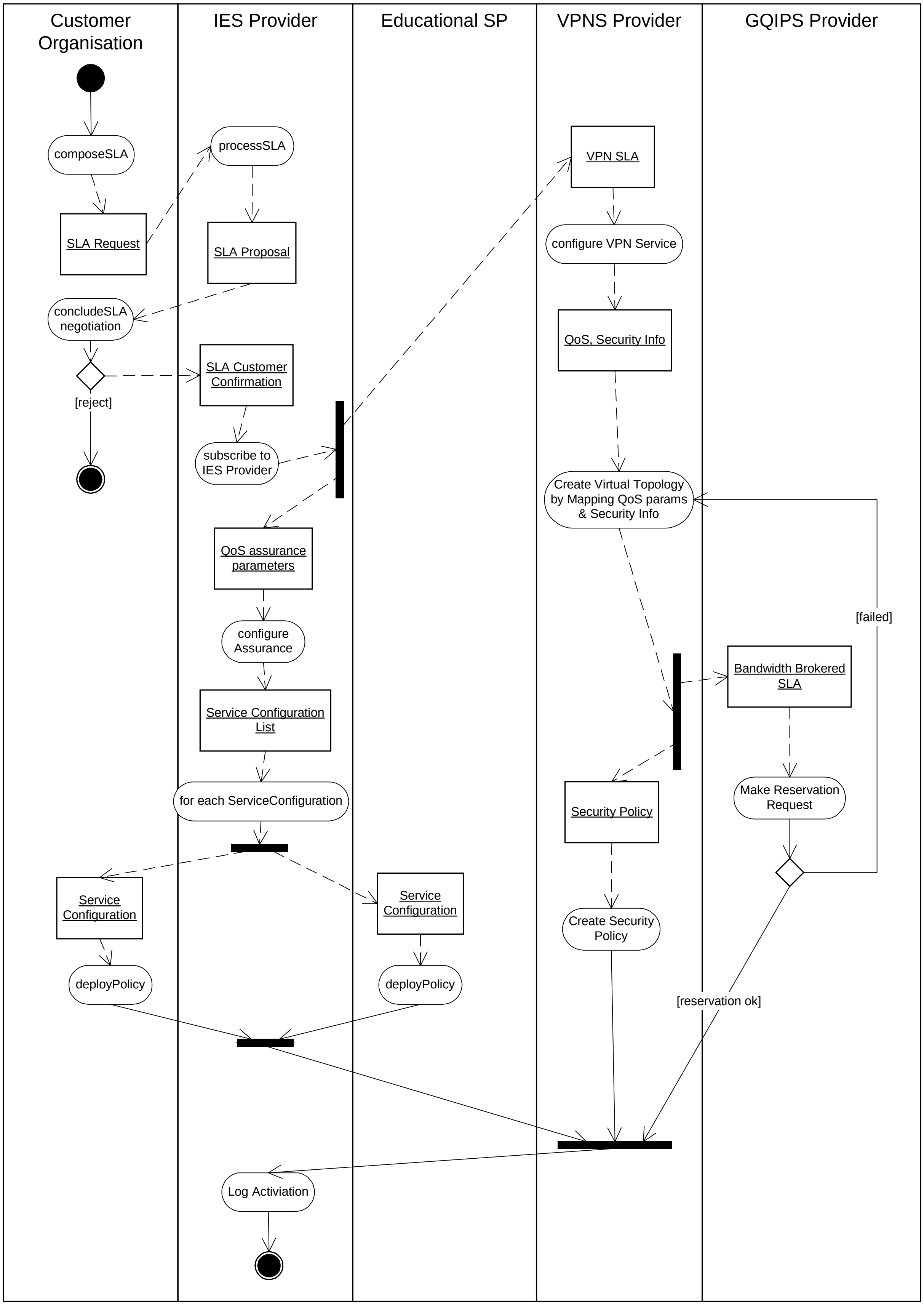
This workflow involves the modelling of system activities, their control flow and their information flows, which occur in the management system processes to be developed. These management processes are identified from the use cases defined in the previous workflow.

The models produced by the workflow are:

- Management System Process activities and Control Flows captured as Activity diagrams (graphs). In UML v1.4, these activity diagrams can also represent information objects, which are passed between the system activities.
- Optionally, a collaboration diagram with analysis objects representing system activities (as defined in the system activity diagrams) and objects from the class diagrams representing data flows between these analysis objects. Some developers prefer this representation, however, it does not make explicit the control flow decisions which govern the interaction between the system activities.

6.3.1 Example: System Process for Fulfilment Assurance System

If we continue the fulfillment-Assurance example from previously, this workflow would indicate that the next development task is to develop a system activity model for Fulfilment-Assurance. This system activity diagram needs to show the system activities, control flow, information objects, branching, iteration and synchronization points. The swimlanes are used to represent the different roles in the management processes. Flows across the swimlanes are used to indicate control or data flows across organizational (role) boundaries. The level of granularity of the system activities are at a relatively high level of abstraction and the information flows are based on information either already specified in the External Information Model of the ODF or are based on some appropriate standard information model e.g. DMTF, TMF. The modeling of the system activity diagram(s) requires several iterations to ensure consistency with appropriate information models and activity granularity etc.


Customer Organisation
IES Provider
Educational SP
VPNS Provider
GQIPS Provider
composeSLA
processSLA
VPN SLA
SLA Request
SLA Proposal
configure VPN Service
concludeSLA negotiation
QoS, Security Info
SLA Customer Confirmation
[reject]
subscribe to IES Provider
Create Virtual Topology by Mapping QoS params & Security Info
QoS assurance parameters
[failed]
configure Assurance
Bandwidth Brokered SLA
Service Configuration List
for each ServiceConfiguration
Security Policy
Make Reservation Request
Service Configuration
Service Configuration
Create Security Policy
deployPolicy
deployPolicy
[reservation ok]
Log Activiation

6.4 Re-model System Processes and Map to Building Block Contracts workflow; Map System Process data to Boundary Information Model(s)

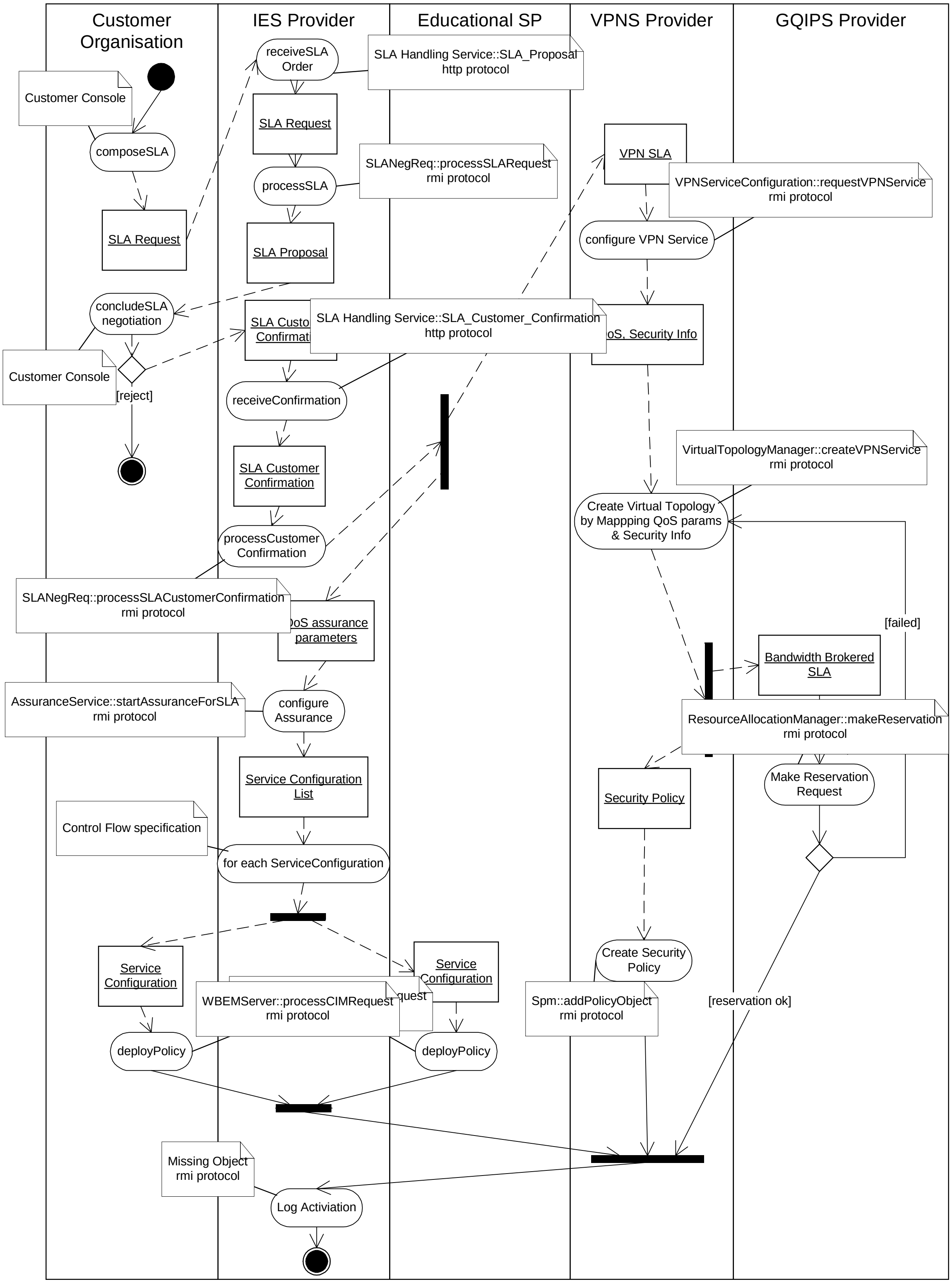
This workflow uses the activity definitions and diagrams to identify candidate Building Block Contracts, which offer equivalent/appropriate behaviours. However it is unlikely that exact matches of behaviour and the appropriate level of granularity will be readily available. Therefore a re-organisation of the system development processes is necessary to decompose system activities to the granularity that matches some of the Building Block Contract behaviours. It is unlikely that all activities of a system process will be performed by Building Block Contracts. These missing functionalities are identified for later development effort in the guideline.

However, behavioural re-arrangement is not the only cause for reorganising the System Processes. Because of the likely mismatch between the chosen Building Block contracts' Boundary Information Models and the information flows in the system process(es), extra activities which perform data retrieval, transformation or generation may also be required. This will make the reorganised system process control flow consistent with the information flow within the system processes. This remodelling work will need to be iterated a number of times until a satisfactory balance between reuse of Building Block Contracts and development of new software (i.e. missing functionality) is achieved.

The result of this workflow is:

- A new set of system activity diagrams, in which each activity is noted as being supported by a (named) Building Block Contract or is noted as not supported.
- Again, optionally, collaboration diagrams indicating information flows between the analysis objects each of which represents a Building Block Contracts in the catalogue, or a new object which needs to be implemented to support the management processes can also be modelled.

6.4.1 Example: System Activity Diagram with Building Block Contract Annotation



6.4.2 Example External Information Model for Fulfilment-Assurance

In our case study, several information objects need tobe passed between the BB Contracts. The workflow identifies these information objects and those that are passed between the Building Block Contracts that are involved in the process. The first four information objects passed between the BB Contracts are SLA_Reqeuest, SLA Proposal, SLA_Customer_Conformation and Assurance_SLA. These obejcts are defined in the External Information Model for the ODF.

6.5 Model Missing Objects and Information Workflow

This workflow supports the modelling of the objects and information which are not supported by Building Block Contracts. These analysis objects are first identified in the system activity diagram (representing the system processes). Some of these analysis objects may be coalesced into one or more analysis objects and, where they interact directly with each other, packaged into a subsystem. The workflow follows standard software analysis and design activities for the modeling of the (classes and components) to support these objects is the performed.

6.6 Implement Building Block Integration

The objective of this workflow is to facilitate the implementation of the necessary ‘application integration logic’. Such integration implements the control and data flows represented in the system activity diagrams. The integration allows the invocation of Building Block Contract interfaces and the bespoke object interfaces (developed in the previous workflow).

This integration logic can be hand crafted in a Business (logic) object. In some circumstance, where the tool used to model the system activity diagrams supports some standard process description languages e.g. XMI, this business object can be (semi) automatically generated. If a workflow engine integration approach is used to integration the Building Block Contract Interface invocations, then the ‘process rules and data flow rules’ for the process automation engine can be generated.

6.7 Map Building Blocks Contracts to Building Blocks & deploy BBs and Business (Logic) Object(s)

This workflow facilitates mapping of Building Block Contract onto Building Blocks. The Building Blocks may be technology dependent and therefore the necessity of using technology or protocol gateways may be required to allow building blocks to interoperate.

The result of this workflow is a full specification of the management system components as it cooperates to support the management processes. The artifacts produced by this workflow will include:

- Activity diagrams where each activity is a component in the management system. The activity diagram will show the control flow of the system management process.
- Class/object Model which models the information passed between the system components
- Collaboration Diagram with each component in the system modelled including the passing of information between the components.

6.8 Perform Testing and Deployment Workflow

This workflow defines and executes the testing necessary for the management system. The workflow involves the design and implementation of test plans as well as their evaluations. The workflow also involves planning and assembly of packages so the system can be deployed onto the target environment. This involves the planning and execution of the deployment of the Building Blocks and other object implementations to ensure the proper execution of the system.

7. Summation

The document has presented the BB Development Guideline and the Business Process Driven System Development Guideline as part of the ODF. Deliverable D11 contains Building Block Contract and Building Block descriptions. These descriptions are presented in such a way so as to illustrate the results of each of the workflow in the Building Block Development Guideline. D11 also provides descriptions of Business Process Driven Systems. Again these systems are described in such a way as to illustrate the results of the workflows of the Business Process Driven System Development Guideline. The evaluation of the methodology as used in the FORM trials and MCG work is described in Deliverable D10.

8. References

[Jacobson 99] The Unified Software Development Process, I Jacobson, G Booch, J Rumbaugh, Addison Wesley, ISBN 0-201-57169-2

[D9] FORM IST-1999-10357/WIT/WP3/1019-v1.4, Deliverable 9 FORM Project Report; Open Development Framework

[Fowler 97] M Fowler, K Scott, UML Distilled, Applying the Standard object Modelling language, published by Addisson Wesley 1997

[GB909]	TeleManagement Forum, GB909, 1999
[Kruchten 2000]	‘The Rational Unified Process – an Introduction’,2nd edition, Philippe Kruchten, Addison-Wesley ISBN 0-201-70710-1
[TMF]	TeleManagement Forum Telecoms Operations Map, GB910, April 1998
[SWEBOK2000]	Guide to the Software Engineering Body of Knowledge, http://www.swebok.org

9. Glossary

Building Block A Building Block (BB) is an atomic unit of software deployment and software management. A BB implements a number of Contracts that are the sole medium for inter-BBs interactions.

Building Block Contract Building Block Contract is an interoperability specification which provide the only means of interacting with a BB. A Building Block Contract specification contains a grouping of interface signatures, information models and interaction behaviours, which can be re-used to support telecommunications management business processes.

Building Block Development Guideline: The Building Block Development guideline describes a *development process, which facilitates the development of Building Blocks and Building Block Contracts.*

Business Process: This is a workflow, which describes (telecommunications or application) business activities, which have to be performed to achieve a business goal.

Management System Process: This is a workflow, which describes management activities, which have to be performed to achieve a management goal. Typically this term is used to identify a sequence set of management activities, which are to be supported by computerisation.

(Telecommunication) Management Business Process: This is a workflow, which describes (telecommunications or application) management activities, which have to be performed to achieve a management goal. Typically this term is used to identify a sequence set of management activities, some of which are to be supported by the Building Blocks in the Open Development Framework.

Package: A general-purpose mechanism for organising elements into groups. Packages may be nested within other packages. All kinds of model elements and diagrams can be organised into packages. Special kinds of packages are model, system and system.

Phase (or Development Phase): A Phase represents the time between two major project milestones during which a well-defined set of objectives is met, artifacts are completed, and decisions are made to move into the next phase. Each phase has a set of workflows, which determine and sequence development activities. Each Phase may iterate these workflows several times to allow artifacts to be refined.

Subsystem: A subsystem is a grouping of classes or other subsystems. Subsystems can be devised either bottom-up or top-down. When working bottom-up, subsystems are suggested based on classes already found. Working top-down means that high-level subsystems and their interfaces are identified before the classes are identified.

Workflow: A workflow identifies a set of activities and their sequencing to achieve a particular goal.

Workflow Process: This consists of the sequence development activities required to achieve a particular development goal. It can also consist of other Workflow Processes

Boundary Information Model: This is the information model, which is communicable across a Building Block Contract. This Information Model is therefore the information which is externalised by the Building Block Contract and which is available (either as input or output) to any user of the Building Block Contract.

Reference Point: A Reference Point is the relationship, which is modelled at the Business level between business management processes, which reside in different Business Roles. They indicate a level of interaction typically between two organisational units and their respective processes. This is important in modelling B2B domains.

Annex 1 Relationship Between the Building Block Guideline and Rational Unified Process

Technical Approach for Building Block Development Guideline

The approach taken in developing the Building Block Development Guideline was to re-use current best practice in software development and to customise and add features or artifacts where required. Therefore the Guideline was not devised from scratch, but rather constructed from the most widely accepted methodologies and then enhanced to suit its needs. Several candidate software development processes were identified [SWEBOK2000], but it was decided to base the 'Context Modeling' workflows loosely on Rational's Unified Process (RUP). Several reasons underpinned this choice:

(i) RUP is widely adopted in the Object Oriented software development community.

(ii) RUP employs the Unified Modelling Language, which is a modelling notation adopted widely in industry and by many standardisation for a in relevant areas such as management, e-commerce and distributed computing

(iii) RUP is a 'development process framework' and thus it is intended to be customised for the development of different specialised artefacts and processes. The provides some flexibility needed to adapt RUP to the particular methodological requirements of the ODF stakeholders

(iv) RUP claims to support component oriented as well as Object Oriented software development.

Therefore some of the Building Block Guideline workflows are a customisation and enhancement of known industrial software methodologies. However it has been augmented with best practices based on experiences in previous research projects and academic work. The Building Block Development Guideline focuses on the Development of Building Blocks rather than the mapping of such models into specific technologies or computing platforms. It also facilitates the specification of Building Block Contracts. A Building Block Contract specifies a grouping of information and behaviours, which can be re-used to support management business processes. A Building Block Contract can be supported by one or more Building Blocks.

Rational Unified Process (RUP)

The Rational Unified Process (RUP) is a software engineering process developed and marketed as a product by Rational Software. RUP is itself a specialisation of the Unified Software Development Process (USDP). RUP is delivered online using Web based technology and consists of more than 1000 hyperlinked pages of text and graphics. It provides a proven disciplined (industrial) process for assigning tasks and responsibilities within a development organisation to design applications and enterprise systems. RUP aims to capture many of the best practices in software development and then attempts to present them in a form that can be tailored for a wide range of projects.

This section briefly identifies the important aspects of RUP, which are utilized later in describing the Building Block Development Guideline. A more comprehensive overview of important aspects of RUP is contained in [Kruchten 2000] and a broader description of UML based development process is also presented in [Jacobson 99].

RUP depicts software development in two dimensions, Phases (Inception, Elaboration, Construction Transition) and Process Workflows (i.e. development activities), which are conducted within each phase. RUP also identifies three Supporting Workflows, which support the co-ordination of the overall development effort called Change/Configuration Management, Management and Environment). Figure 1 represents the four phases, the process workflows and also provides an indication of the level of effort devoted to each process work within each phase.

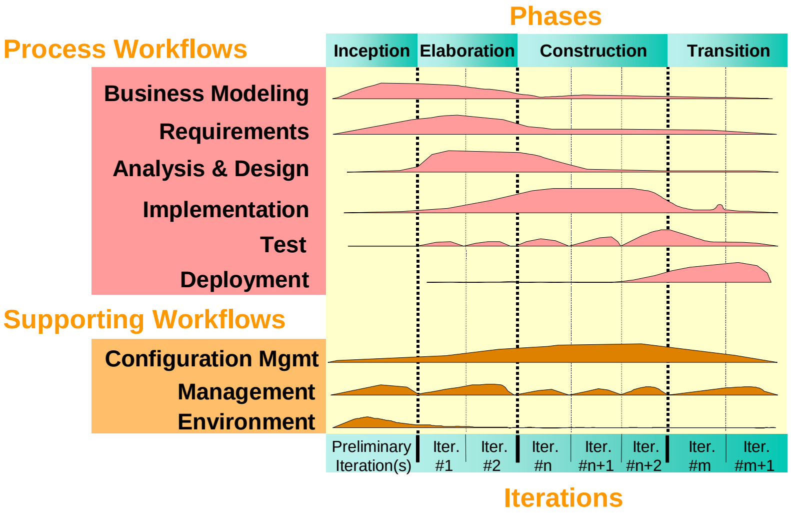


Figure 1 Rational Unified Process Lifecycle Model

RUP is an iterative process in that multiple iterations of the process workflows are expected within each phase. The precise number of such iterations is dependent on the complexity of the solution being developed and the operating context of the development effort e.g. the experience of developers, complexity of application area etc.

The goal of the Inception Phase in RUP is to develop the business case to the extent necessary to justify launching the project [Jacobson2000]. In this phase the workflows determine the scope of the system to be developed as well as a developing parts of the models that would be necessary to support a proof of concept prototype. However the business modeling, requirements and analysis workflows are the principle areas of effort, whereas the implementation, test and deployment workflows concentrate on planning activities and infrastructure selection.

The Elaboration Phase has several specific targets, namely the capturing of 80% of the required use cases of the system under investigation, commencement of detailed design work, completion of a deployment model for the envisaged system, and completion of about 10% of the implementation work completed. The Construction Phase should achieve a complete system implementation ready to begin transition to a user community. The Transition Phase involves the deployment of the completed system into its intended user community and the performance of minor fixes and some fine-tuning.

Therefore the Phases in RUP define ‘When’ in the software development lifecycle activities should be performed. The ‘Who’, ‘What’ and ‘How’ of the development process are defined within the Process and Support Workflows. Thus in each of the workflows, workers or roles indicating who should carry out a development activities are identified, how development activities should be performed and what artifacts (models, design elements) are required to be developed. The Process Workflows defined in RUP are:

(i) Business Modeling,
(ii) Requirement Management,
(iii) Analysis & Design,
(iv) Test,
(v) Deployment

and the Supporting Processes are:

(i) Change/Configuration Management,
(ii) Project Management and
(iii) Environment.

Relationship between the BB Development Guideline and Rational Unified Process

Implementing a system based on RUP can be achieved by authoring and adhering to a development case, which defines the specialisations/additions to the original RUP phases, activities, process workflows and artifacts. The Building Block Development Guideline can be thought of as such a development case, which facilitates the development and specification of Building Blocks and Building Block Contracts. In particular the development case focuses on tailoring the various aspects of the RUP’s business modeling, requirements modeling, analysis & design, implementation and deployment workflows.

The Context Modeling Phase of the BB Development Guideline can be thought of as a filter on RUP that selects and customises the activities that are to be performed and artifacts that are to be produced as well as adding additional elements where necessary. For the Open Development Framework, it was found that RUP was lacking in support for a “component design”. In RUP, a component is a runtime unit of software, whereas in FORM the desire is to provide a reusable grouping of models and artifacts, which is rich enough to adequately represent the properties, pre- and post- conditions, constraints, potential usage, and interfaces of a Building Block. This grouping is termed a Building Block Contract. A Building Block may support one or more Building Block Contracts. A FORM Building Block more closely resembles a “Sub System” in RUP.

This Guideline customises the RUP phases to reflect the goal of developing Building Blocks and Building Block Contracts for the FORM Open Development Framework.

Mapping the Guideline Development workflows into an extended RUP

The Building Block Development Guideline can be presented as an extension and customization of several aspects of RUP. RUP was designed for a broad range of application and enterprise systems. However this guideline is not focused on application development, but rather reusable component development. Therefore augmentation and extension of various parts of RUP were necessary because of the development target. For example, because the guideline is focused on the development of management components, considerable effort is needed to initially model the problem domain in order to identify and scope candidate components as well as to design and specify the components (once they are agreed). Also, management systems (for service and network management) are frequently distributed across organisational or administrative boundaries. Such management system boundaries are usually represented in the telecoms area as reference points[4]. Such reference points then become sets of coarse-grained integration points for (possibly multi-vendor) software components. Such reference point specifications are not commonplace in other software markets and domains and hence are not currently part of RUP. The customisations of RUP included changes to *Development Phases*, *Model & Artifact Selection*, *Process Workflows* and *Documentation*.

Development Phases & Iterations.
The Context Modeling Phase for this Guideline is more extensive than would normally be the case for general application of the Inception Phase of RUP.. The Business Modeling work is very important in initially scopeing the management process areas and domain responsibilities for which the Building Blocks will provide support. The Business Modeling work involves identifying the domain boundaries of the envisaged management systems and the organisation in which they may reside, identifying management use cases and initially modeling the management processes which support them. An additional outcome of this phase is the modeling analysis objects, which support these use cases and management processes. These analysis objects are used in the Building Block Development Phase in the identification and determination of candidate Building Blocks. Also this phase allows the modeling of reference points between Business Organisations identified during business modeling. External influences in this phase include the adherence/alignment with relevant telecom industry standards e.g. TeleManagement Forum’s Telecom Operation Map (and F-A-B processes) [TMF]. Such

[4] Different standards bodies have used the notion of reference points to assist in describing relationships between either business entities or systems e.g. TINA.

standards can have a significant influence on the management processes being modelled as well as on the information (object) model(s) associated with these processes. However, it is important to remember that the objective of this development Guideline is the development of Building Blocks and Building Block Contracts. The effort therefore in this (inception) phase is to identify the business processes, actors, potential organisational boundaries, etc. which impact on and place in context the building blocks and building block contracts which will be subsequently developed in the later development phases.

The Building Block Development Phase focuses on the development of specific building blocks and Building Block Contracts and involves testing of these Building Blocks. This involves the identification of Building Blocks and the re-organisation and grouping of analysis objects into re-usable Building Block designs and specifications. The elaboration phase also involves the modeling of information used both within the building block(s) as well as those (information objects) passed into or communication out of the Building Blocks. The information communicated by a Building Block Contract is termed its ‘Boundary Information Model’ and is documented an explicit information model indicating the informational requirements and outputs of the Building Block. The key output of this phase is the specification of Building Block Contracts and the design of Building Blocks, which support them.

The Construction Phase and Transition Phase is not considered within the BB Development Guideline.

Models and Artifacts Selection.
RUP defines a very large set of potential modeling and documentation artifacts. The BB Development Guideline prescribes a reduced set of models necessary to design and specify Building Blocks. Also the guideline specifies a Building Block Specification Template, which presents a rich description of the Building Block, essential for later reuse.

Two additional artifacts, not in RUP, are contained in the Guideline, namely:

(i) An explicit architectural (structural) diagram, called the Reference Architecture, which indicates reference points (boundaries) between management/administrative boundaries and the placement of business processes within and/or across these boundaries. This architectural diagram attempts to reconcile the management process areas (for which building blocks are to be developed) with the organisation or administrative boundaries being considered within the Business context.

(ii) The explicit modeling of Building Blocks and specification in XML of the Building Block Contracts. These Building Block models are considered fundamental to the FORM Framework and do not have a direct equivalent in RUP. The most similar package of models defined in RUP, which includes the grouping of such required models, is that of a Sub System package.

Process Workflows
The Guideline concentrates on the Process Workflows (rather than the support workflows). This does not mean such support workflows are not relevant, on the contrary they are very useful. However, as this guideline is focused on the actual development

activities (rather than their management or product deployment), the Process Workflows are the only activities customised. This customisation is necessary for the development of Building Blocks and the modeling artifacts that are used to specify them.

Documentation
The Guideline focuses on only a few key document templates, including Vision Document, Software Architecture and Building Block Specification.

Terminology used in the Guideline
The concepts and terminology typically used to describe methodologies for software design can be confusing. This section provides some simple definitions of the various terms used by the Guideline. The Building Block Development guideline describes a *development process*. A development process consists of a *set of phases*, which provide major checkpoints during the development. Each phase has associated with it a set of *workflow processes*, which determine and sequence *development activities.* Each Phase may iterate these workflow processes several times to allow the design models to be refined. Later Phases expend much greater effort on software implementation issues whereas implementation issues, when addressed in earlier phases, tend to focus on infrastructure planning and proof of concept. The Glossary contains complete definition of key terms used in the Guideline.