ARIS meets RUP
The ARIS Unified Information System Development Process

Martin Plümicke
University of Cooperative Education Stuttgart
Department of Information Technology
Florianstraße 15
D–72160 Horb
tel. +49-7451-521142
fax. +49-7451-521190
m.pluemicke@ba-horb.de

Abstract: Starting from the Rational Unified Process RUP and the ARIS life cycle model the ARIS Unified Information System Development Process AUP is designed. Similar as the RUP is developed from the classic waterfall model as an iterative and incremental process to build software, the AUP is developed from the ARIS life cycle model as an iterative and incremental process to build business information systems. As software development is a part of the information system development the RUP is integrated in the AUP. The modeling language for the AUP is a combination of UML and diagrams from the ARIS framework concept.

1 Introduction

In last years great efforts are done to improve software development processes [Ba00]. Starting from the traditional waterfall model many other models, as V–modell, prototyping–model, and The Rational Unified Process, are developed. These models consider almost only the software development.

In the same time the application of computer science in the business were extended from pure software solutions to large information systems, which implement complete business processes. For this beside software implementation, databases, networks, and its integration must be considered.

Software-engineering is presently almost done object–oriented. The Unified Modeling Language (UML) [FS00] defines many different model types to describe a software product from different views. UML itself is no software engineering process. For UML a software engineering process is developed by Rational Software: The Rational Unified Process (RUP) [JBR98, Kr03, RUP]. RUP differs in two points essentially from traditional software engineering processes like the waterfall model. On the one hand RUP has no static phases as analysis, design, implementation, and test. There are different workflows, which are applied iteratively. The main advantages of iterative software development
Figure 1: ARIS life cycle model
is that aberrations are detected earlier, which can reduce costs enourmously. On the other hand RUP is a configurable process, which means that RUP can be varied to accommodate different situations.

The ARchitecture of integrated Informations Systems (ARIS) [Sc02, Sc01] is a framework concept to describe a complete business information system. ARIS is extended by a development process, the ARIS life cycle model, which is similar to traditional software engineering processes.

The content of this paper is the advancement of the ARIS life cycle model to a modern iterative process, which is enhanced from the RUP.

In this paper we will start in the next section by describing RUP and ARIS. In the third section we consider some related work, which deals with the integration of ARIS and UML. Then, in the fourth, the main section, we define the ARIS Unified Information System Development Process (short: ARIS Unified Process, abbreviated as AUP). Finally we close with a conclusion and an outlook.

2 Overview over ARIS and RUP

2.1 ARIS

The ARIS framework concept considers business information systems by five different views.

Function view: In the function view all functions are modeled. Functions are defined as transformations from input products to output products.

Organization view: In the organization view the structure of the company is modeled.

Data view: In the data view the data are modeled which are input or output products, which are environment data, and which are events or messages in the control view.

Output view: In the output view products are modeled, which are either produced or bought for ennobling.

Control view: In the control view on the one hand the connections between the different views are modeled. This is the static part of the control view. On the other hand the dynamic part of the business process is modeled. This is done by the Event–Driven Process Chains (EPCs) [KNS92].

For the building of a business information system each of these five views are considered by the ARIS life cycle model (figure 1).⁴

⁴The figure is taken from www.iwi.uni-sb.de as a translation of a figure in [Sc02].
Figure 2: Rough ARIS process
The five phases of the ARIS life cycle model are the basis for the workflows of the ARIS unified process in section 4. The rough ARIS process for modeling a business information system is shown in figure 2. This life cycle model is very similar to the waterfall model. The main problem of the waterfall model is, that the risks of the system development are consequently shifted in later phases. But fixing of detected defects becomes during the software process more expensive. This means the risks should be managed as early as possible.

The ARIS framework concept and the life cycle model is summarized in the ARIS house (figure 3). In the ARIS house only the second, third, and fourth step of the ARIS life cycle are considered. The further steps are considered together for all views of the information system.

For each phase of each view there is at least one diagram type to model the view of the phase. The control view is divided in a static and a dynamic part. In the static part for each connection of different views there is a diagram type to model it. The dynamic part is modeled by the EPCs. We do not present these diagram types at this point of the paper as the diagram types are given during the description of the ARIS Unified Process in [8c02].

---

2The figure is taken from www.iwi.uni-sb.de as a translation of a figure in [8c02].

3The figure is taken from www.iwi.uni-sb.de as a translation of a figure in [8c02].
In the framework of this paper we do not consider the output view. The reason is, that the development of a business information system either only supports the product development or the products are data. In the first case the information system supports the production in some production steps. But these steps are defined in the function view. In the second case the products are considered in the data view.

2.2 RUP

The main characteristics of the Rational Unified Process RUP\(^4\) is that it is iterative and incremental. This means that large software development projects are broken down into a number of smaller mini projects, which are easier to manage. The key point is that each iteration contains all elements of a standard software development project: planing, analysis and design, construction, integration and test.

This property leads to a two dimensional description of the process (fig. 4)\(^5\):

- The horizontal axis represents time. It is the dynamic aspect of the process. The process is divided in phases and iterations.
- The vertical axis represents the static aspect of the process: the workflows, the activities, the workers, and the artefacts.

---

\(^4\)The content of this section is taken from [RUP, JBR98, Kr03]

\(^5\)The figure is taken from [RUP].
The diagram itself shows how much work is done of which workflow in the different phases.

2.2.1 The dynamic structure of the RUP – the phases

The software development project is divided in four phases. Each phase is again divided in several iterations. Each iteration is closed by a milestone and each phase is closed by a major milestone.

In the description of the different phases we will refer to the workflows, the workers, and the artefacts, as they are orthogonal to the phases. These aspects are considered in the following sections.

Inception Phase: The purpose of the inception phase is to get a base for the project. This involves the following aspects:

- establishing feasibility (e.g. prototyping to validate the business requirements)
- creating business case to deliver quantifiable business benefit
- capturing essential requirements to help scope the system
- identifying critical risks

The primary emphasis is on the requirements and the analysis workflow. However some design and implementation as prototypes also might be done. The test workflow is not applied, as the only software artefacts are prototypes, which are later thrown away.

The primary workers of the inception phase are the project manager and the system architect.

The milestone is the Life Cycle Objective, which includes the following conditions, such the milestone is reached: stakeholder have agreed the project objectives, system scope have been defined (10% – 20% of the use-cases), key requirements have been captured, costs and risks have been assessed, confirmation of feasibility, and the architecture is outlined.

Elaboration Phase: The purposes of the elaboration phase are summarized as follows:

- create an executable architectural baseline
- refine the risk assessment
- capture use-cases to 80%
- create a detailed construction plan including resources and costs

The focus of the elaboration phase is the creation of an executable architectural baseline. While in the design workflow a stable architecture is created, in the implementation workflow the architecture baseline is constructed. This is no prototype, rather a first cut out of the design system.
The milestone is the *Life Cycle Architecture*, which includes the vision document, an executable architectural baseline demonstrates that the important risks are identified and resolved, a realistic project plan with statements of time, money, and resources, and an agreement with the stakeholders to continue the project.

**Construction Phase:** The purposes of the construction phase are given as:

- complete the requirements
- finish the analysis model
- finish the design model
- evolve the architectural baseline from the elaboration phase into the final system

The focus in this phase is on the implementation workflow. Testing becomes also more important.

The milestone is the *Initial Operational Capability*, which includes the software product, the UML model, a test suite, and a user manual. The actual expenditures respectively the planned expenditures are acceptable. The stakeholders have agreed to the transition to their environment.

**Transition Phase:** The transition phase starts when beta testing is completed. The purposes can be summarized as follows:

- correct defects
- prepare the user site for the new software
- prepare documentation
- provide user consultancy

The emphasis is on the implementation and the test workflow. Hopefully, by this point, there should be very little work being done in the requirements and the analysis workflow.

The milestone is the *Product Release*, which includes the software product, the actively using of the product by the users, the user support plans, and the user manuals.

Each phase can be broken down into iterations. An *iteration* is a complete development loop resulting in a release of an executable product, which grows incrementally to become the final system. The benefit of the iterative approach is, that risks are migrated earlier, changes are more manageable, there is a higher level of reuse, and the quality is better overall.

**2.2.2 The static structure of the RUP**

The static structure is represented by using four modeling elements: the *workers*, who is doing something, the *activities*, how it is done, the *artefacts*, what is done, and the *workflows*, when it is done.
The workers

A worker defines the behavior and responsibilities of an individual, or a group of individuals working together as a team. In the RUP the worker is more a role defining how the individuals should carry out the works than the individuals themselves.

The activities

The activities are asked to perform by a specific worker. An activity is considered as creation or updating an artefact. For example plan an iteration is assigned to the project manager or find use–cases and actors is assigned to the system analyst.

The artefacts

An artefact is a piece of information that is produced, modified, or used in process. Artefacts are used by a worker as input for an activity, and as results of the activity. Artefacts are for example models (use–case model, analysis model), documents, source code, and executables.

The workflows

In the RUP the workflows (fig. 4) are divided in two groups: the core process workflows and the core supporting workflows. In this section of the paper we will give only a short overview of the workflows, as in section 4 the workflows of the AUP, which are defined as an extension of the RUP workflows, are considered more detailed.

Business Modeling: One of the major problems is that the software engineering and the business engineering community do not communicate properly with each other. The purpose of the business modeling is to provide a common modeling language for both communities. The business processes are documented using the so called business use–cases. This workflow is one point to connect the RUP and the ARIS concepts.

Requirements: The purpose of the requirements is to describe what the system should do. The developers and the customers should agree to the description, which is called the vision document. The actors/users and the use–cases are identified. These are documented in the use–case model. The use–case model is used as a baseline during the workflows analysis & design and test.

Non–functional requirements are described in the supplementary specification.

Analysis & Design: The purpose of the analysis & design workflow is to show how the system will be realized. The result is the design model. It is an abstraction of the source code and consists of design classes structured into design packages and design subsystems. It also contains descriptions of how objects of these design classes collaborate to perform the use–cases.
Implementation: The system is realized through implementation of components. It is described, how to use existing components and implementing new components in way, such they are reusable. Implementation languages are for example Java, C++, or Ada.

Test: The focus of the test workflow is to ensure the quality of the software. The purposes of testing are:

- To verify the integration between the objects
- To verify the proper integration of all components of the software.
- To verify that all requirements have been correctly implemented.
- To ensure that all defects are removed before the software is deployed.

As the RUP is an iterative process, in each iteration the different releases are tested. This allows to detect defects as early as possible, which radically reduces the cost of defects fixing.

Deployment: The purpose of the deployment workflow is to successfully deliver the software to its end users. This includes producing external releases of the software, packaging the software, distributing the software, installing the software, and providing help and assistance to the users.

Although deployment activities are mostly done in the transition phase, many preparation have been done in earlier phases.

Project management: In the RUP the project management workflow has three purposes:

- A framework for managing software-intensive projects.
- Practical guidelines for planning, staffing, executing, and monitoring projects.
- A framework for managing risk.

Some usual aspects as staff management, budget management, or contract negotiations, of project management are not considered in the RUP.

The planning of an RUP software project is done on two levels: phases planning and iteration planning. The plan of the phases is very rough, on the other hand the iteration plans are very detailed.

Configuration & Change management: In this workflow the control of numerous artefacts produced by the many people who work on the project are described. The following problems are considered: simultaneous update, overall notification of changes, version management.

Environment: The purpose of the environment workflow is to provide

- the software development tools
- configuration of the tools
- configuration of the process (cp. section 2.2.3)
• process improvement
• training
• technical support, administration, backups, etc.

Some activities are closely connected to the **RUP** configuration, which is described in the following.

### 2.2.3 **RUP** configuration

The **RUP** is such global, that it can be used without any configurations and changings. In particular this holds for small and middle software companies, without an explicit culture for the processes. On the other hand the **RUP** is a framework, which can be modified, adapted, and extended for special requirements of the company. If the **RUP** is configured, soonest the workers or the activities are changed.

### 3 **Related work**

In this section we present some papers about **ARIS** and **UML**. The goal of all approaches is to improve **ARIS** by integration of some object–oriented models. Nearly no paper attend to the process itself, as our approach in this paper.

As described in section 2.1 in [Sc01] the **ARIS** framework concept and its life cycle model is presented. This is a method to describe the building of business information systems. The approach is based on the traditional software development paradigms like the waterfall model, although in some views object–oriented models are integrated.

There are some papers, which give approaches to integrate the business views to object–oriented software development methods. Often these approaches describe the integration of Event–Driven Process Chains (EPC) into the Unified Modeling Language (**UML**) [FS00].

In [LF01] on the one hand the connections between the different workflows of the **RUP** and the EPCs are described. On the other hand a mapping between different **UML** diagrams and EPCs are given.

In [NTZ98] two approaches are described to combine process and object–oriented modeling. The first approach gives a transformation of business process models (EPCs) into object–oriented models. The **UML** models **use–case diagram**, **activity diagram**, and **class diagram** are considered.

The second approach summarizes the ideas of integration both methods by object–oriented event–driven chains (oEPC), which are introduced in [SNZ97]. Within oEPCs business objects and events respectively rules are defined as classes.

In [LA98] the base of the considerations is the **ARIS** framework concept, similar as in our considerations. Besides the regard of the connection between EPCs and the different **UML** diagrams, which build the basis of the considerations in [LF01], in this paper the statement
is given that a straightforward sequence as in the waterfall software development process is not adequate to build a business information system. This statement is the base of our approach.

4 The ARIS Unified Information System Development Process

In this section we will extend the RUP, which is a method to develop iteratively a software system, to an iterative process developing business information systems. We call this process The ARIS Unified Information System Development Process (short ARIS Unified Process abbreviated as AUP). In our approach the rough ARIS process to model business information systems (figure 2) is developed to the iterative and incremental process AUP as the traditional waterfall model is developed to the iterative and incremental process RUP. As the development of a business information system contains software development, the RUP as described in section 2.2 is part of the AUP.

In this section we will describe the AUP in a similar way as we outlined the RUP above. After the overview, we will present the dynamic structure, which is given along the time in the phases strategy, inception, elaboration, construction, and transition. Then we give the static structure, which consists of the workers, the activities, the artefacts, and the workflows.

We will only consider the core process workflows as the core supporting workflows of the RUP are unchanged in the AUP.

Before we present the dynamic and the static structure, we will give an overview, which shows the two orthogonal structures.

In figure 5 the phases and the different workflows are presented. There are five main workflows. The first one is the strategy workflow. The complete activities of this workflow are done during the strategy phase. The other four workflows requirements definition of control flow, requirements definition, design specification, and implementation are derived from the rough ARIS process (figure 2). The workflows requirements definition, design specification, and implementation are divided into subworkflows. For each view of the ARIS–house there is respectively one subworkflow.

We close the overview by a description of the integration of the RUP into the AUP. In comparison to the RUP, the workflow requirements definition of control view is comparable to the business workflow. The main difference are the assigned diagrams. As in the RUP business workflow, the business processes are modeled only by different use–case diagrams and business process object models, in the AUP the business ideas are modeled additionally by the EPCs. The subworkflow requirements definition function view and the first step of the subworkflow data view, the macro view, are comparable to the RUP workflow requirements. The subworkflow design specification function view and the subworkflow control view are comparable to the RUP analysis & design workflow.
### 4.1 The dynamic structure of the AUP – the phases

As in section 2.2, we will now consider the dynamic structure of the AUP first. An information system development project is divided into five phases. Each phase is again divided into several iterations as in the RUP. Each iteration is closed by a milestone and each phase is closed by a major milestone.

The phases are comparable to the RUP phases. In the AUP there is an additional phase, the strategy phase.

**Strategy phase:** The first phase of the information system development project is the strategy phase. In this phase only the Information System–oriented Strategy Applications Concepts workflow and the Requirements data view subworkflow are active. In the first
workflow the management have thoughts about the company’s strategy, which are influenced by the technical progress. The other active workflow determines the macro description of the used data which corresponds also to the general possibilities of the information technique. For example a phone call or fax are macro descriptions of data.

In these phases the framework of the business information system is determined.

**Inception phase:** The inception phase can be described very similar as in the RUP (cp. section 2.2.1). For the inception phase one principle is very important: It holds *structure follows process follows strategy* [OF96]. In the AUP this principle is applied iteratively. In the strategy phase the company’s strategy is determined. Then, the *requirements definition of the control view workflow (process)* and the *organization workflow (structure)* form iteration by iteration the business model and the organization model.

The other phases *elaboration, construction, and transition* are not considered in this paper as they are similar to the respective phases of the RUP.

### 4.2 The Workflows

In the following we will describe the core workflows *Information System–oriented Strategy Applications Concepts, Requirements Definition of Control View, Requirements Definition, Design Specification, Implementation* and its subworkflows (fig. 5). The core supporting workflows from the RUP are unchanged in the AUP.

#### 4.2.1 Information System–oriented Strategy Applications Concepts Workflow

This workflow is only active in the strategy phase. The workflow is described in the section of the strategy phase.

![Figure 6: Workers and artefacts of the IS–oriented Strategy Applications Concept Workflow](image)

**Workers:** In this workflow the executive board of the company is responsible for the result of the workflow. During the workflow an analyzer of the information technique possibilities is involved.
Artefacts: The result of the workflow is a strategy glossary, which contains the strategy for the information system, which is to build.

4.2.2 Requirements Definition of Control View Workflow

The purposes of this workflow are to understand the structure and the processes of the company, to ensure that the customer, the user, and the developer have the same view of the company, and determine which requirements are necessary to support the company. This workflow is similar to the RUP business workflow.

The requirements definition of control view workflow is considered as an own workflow. In the other workflows the control view workflows are subworkflows. The reason is that the main importance of the control view here is a description of the whole information system in a global view. In the other workflows (design specification, implementation) the control view workflows are only necessary for the interface description of the different other views.

Worker:

- Business process analyst: The business process analyst analyses the consisting company, the structure and their processes. He specifies roughly the optimized processes and the following company’s structure.

- Business process designer: The business process designer takes the roughly defined processes and describes the processes in detail. From this follows the detailed company’s structure.

Artefacts: The results of this workflow are use–case diagrams, a mapping from a planning level diagram to different corresponding data (data level diagram), and perhaps different EPCs. The collection of these diagrams is called the business model (cp. figure 7).

![Figure 7: Workers and artefacts of the business model](image)

The use–cases describe the rough processes and the organization units which are responsible for the respective functions of the processes. The use-cases can be modeled more
detailed by state diagrams and sequence diagrams. The EPCs finally describe the process in detail. In the EPCs (modeled as extended EPCs) all views, organization, function, data, and product are connected. These models together are called the use–case model (cp. figure 8).

The allocation of the different data to the different planning levels are described in the data level diagram.

In comparison to the RUP approach the activity diagrams are not components of the use–case model, as the activity diagrams are less powerful than the EPCs.

In comparison to ARIS we do not model the connection between function and data view workflows by class diagrams at this point. We move this in the design specification workflow. The reason is, that it is often not possible for customers and stakeholders to understand class diagrams. Class diagrams are far away from the modeling techniques of business engineering.

This requirements definition of control flow workflow describes the connection between the three different views on the information system, which is to build. It is obvious, that this can only be done if the views themselves are also modeled. The iterative approach guarantees this, as the corresponding workflows are parallel active in the respective iterations.

4.2.3 Requirements Definition Workflow

The purpose of the requirements definition is to describe what the system should do and allows the developers and the customers to agree on that description.

The subworkflow

The workflow is divided into three subworkflows. Each subworkflow considers another view of the whole system respectively its environment.
• **Organization view:** The organization view workflow forms the structure of the company. This is done in close coordination to the company’s strategy and the process definition.

• **Data view:** The data view workflow forms the macro and the micro description of the data structure. During the strategy phase the workflow starts by modeling the macro description as the macro data objects are defined in the strategy phase. At the end of the inception phase the modeling of the micro description starts.

• **Function view:** The function view workflow models the functionality of the information system.

**Worker:**

• **System analyst:** The system analyst controls and coordinates the requirements discovery. He gives the framework of the whole system and is responsible for the vision document.

• **Function designer:** The function designer specifies in detail all functions of the system.

• **Architect:** The architect is responsible for the identification of the use–cases and the functions, which influence the architecture.

![Figure 9: Artefacts of the system analyst, the function designer, and the architect](image)

• **Organization designer:** The organization designer is responsible for the advanced company’s structure, which follows from strategy and process.

• **Data analyst:** The data analyst have to identify the different data representations of the information system. He is responsible for the macro data description.

• **Data designer:** The data designer takes the macro description and builds a data micro description, which contains the data structure of the information system.
Artefacts: The artefact, which is developed in all subworkflows is the vision document. The vision document contains the general vision of the core project’s, key features, and the main constraints.

The artefacts of the subworkflow organization view are the organigram and the diagram of the process-oriented planning levels. These two diagrams form the organization model.

The artefacts of the data view subworkflow are a model of the macro data description and ERM–diagrams as a model of the micro data description.

Further artefacts are the function hierarchies, which are modeled in the function workflow.

4.2.4 Design Specification Workflow

The purpose of the design specification workflow is to show how the system will be realized in the implementation phase.

The subworkflows

This workflow is divided into four subworkflows. In the design specification the connection between the control view workflow and the other three workflows is closer than in the requirements definition. In the requirements definition the control view is a global view to the whole information system. Here the control view describes only the interfaces between the models of the different other views.

The main activities of the control view, the organization view, and the function view subworkflows start at the beginning of the elaboration phase. The data view subworkflow starts later, as the data view subworkflow of the requirements definition is longer active to model the ERM–diagrams.

- Organization view: In the organization view subworkflow a logical network is designed. The network is defined by its nodes and its edges. It is also modeled the rough capacity of the edges, a mapping of nodes and existing networks to organization units, and the protocols which are used.
• **Data view:** In the data view subworkflow the entity relationship model is transferred into statements of the data definition language SQL. This is done in some steps as normalization, definition of views, and translation in SQL.

• **Function view:** In the function view subworkflow the algorithms of the system’s functionality are determined. UML offers for this sequence diagrams, collaboration diagrams, activity diagrams, and state diagrams. This subworkflow has a close connection to the control view subworkflow, where the classes are modeled.

• **Control view:** In the control view workflow the interaction of the models of the other three subworkflows are built. The class diagrams are modeled as connection of the data and the function view subworkflows. But there is another problem. Often there is no obvious isomorph mapping from database tables to the classes, as the structures of the database tables and the classes are completely different. Therefore a mapping from the class structure to the database schema must be developed.

The connection of the organization view and the data view workflow is described by a detailed authorization concept. The concept is a refinement of the mapping from data to planning levels of the requirements definition. In ARIS at this point there is additionally a description of distributed databases required. In the last years the capacity of networks arises enormously and the encryption techniques became more and more standard. Therefore in new designed systems databases are often hold at central points. In advanced systems the different data sources are made transparent by so-called application servers. This means that at least the collection of different data sources to an application server is to model at this point.

The connection between the organization view and the function view workflow is described by a mapping from network nodes to different program modules. Today it is often not necessary to consider this mapping as the program modules are hold central and the user interface is done by so-called thin-clients as web-clients.

**The workers**

• **Architect:** The architect controls and coordinates the technical artefacts. He is responsible for the matching of the different artefacts built in the different view subworkflows.

• **Class designer:** The class designer is responsible for the realization of the use–case model and the building of the application server design. The realization is done by modeling of classes, packages, and subsystems.

• **Data-object-translate designer:** The data-object-translate designer is responsible for the mapping from the class structure to the database schema.

• **Function designer:** The function designer is responsible for the deployment of the algorithms, which realizes the functions of the system.
- **Database designer**: The database designer is responsible for the transfer of the ERM–diagrams to SQL–statements.

- **Network designer**: The network designer designs the logical network.

- **Authorization designer**: The authorization designer determines the authorization of the different organization units to the different data.

For each of the designer there is also a reviewer, who reviews the artefacts, which are produced in the respective subworkflows.

**The artefacts**

One of the main artefacts is the *design model*. The design model consists of the class diagrams and the additional diagrams (object diagrams, state diagrams, sequence diagrams, collaboration diagrams, and activity diagrams) which describe the dynamic behavior of the class instances. The design model contains also the diagrams which are artefacts of the function view subworkflow. The mapping from the class structure to the database schema and the structure of a possibly given application server is also modeled in the design model. The design model is structured in subsystems and packages.
The **authorization concept** is the artefact which describe the authorization of organization units to work with different data.

The artefact which models the data view is the **database schema**. This is represented in the data definition language **SQL**.

In the organization view subworkflow the **logical network** is the modeled artefact.

### 4.2.5 The Implementation Workflow

The implementation workflow has five purposes:

- To define the organization of code in terms of subsystems.
- To implement classes and databases in terms of components.
- To test the developed components as units.
- To establish the network.
- To integrate into the executable information system the results of the implementers.

In the **RUP** three key concepts are introduced: **builds**, **integration**, and **prototypes**. These key concepts will be extended in the **AUP**.

**Builds**

A build is an operational version of the system or a part of the system that demonstrate a subset of the capabilities to be provided in the final product.

Builds in the **AUP** are parts of the implemented system, done in a programming language as well as parts of the database or parts of the network.

**Integration**

The integration is an activity in which separate software components, database components, or components of the network implementation are combined into a whole. In the **AUP** the integration is done incrementally, which means that the components are built in small pieces and then combined into a working whole by the addition of one piece at a time.

The incremental integration offers the following benefit: Faults are easier to locate, the components are tested more fully, and some parts of the system is running earlier.

**Prototypes**

Prototypes are used to reduce risk. Prototypes can reduce uncertainly surrounding the following issues: business viability, stability and performance of key technologies, understanding of the requirements, look and feel, and the usability.
The subworkflows

The three subworkflows control view, data view, and function view starts at a similar point. The activities in the data view subworkflow become earlier less, as by the result of the design specification, the SQL queries, the whole work is nearly done. In the other two subworkflows work is done nearly parallel, as the implementation of the classes is closely connected to the implementation of the functionality.

In the organization view subworkflow the network is established. The main work starts later as in the other subworkflows. The different nodes of the network can be simulated by only real existing computer. This is the reason, that the establishing of the network will start, when the databases and the implemented code is nearly finished.

The test workflow and the deployment workflow are not considered in this paper, as they are very similar to the respective workflows in the RUP. They must only be added by testing and deploying the databases and the network.

The workers

- **Architect**: The architect defines the structure of the implementation model (packages and subsystems)
- **Code implementer**: The code implementer develops the separate components and tests them.
- **Database implementer**: The database implementer implements the database components.
- **Network builder**: The network builder transforms the logical network to physical one and establishes it.
- **System integrator**: The system integrator constructs a build.

For each of the implementer respective the network builder there is also a reviewer, who reviews the artefacts, which are produced in the respective subworkflows.
The artefacts

The main artefacts of the implementation workflow are the implementation subsystems, the components, the integration build plan, and the physical network.

The implementation subsystems are a collection of components and other subsystems. A component is a source file, a binary, an executable, or a database component, or another file. The integration build plan specifies the builds to be created when the system is integrated. The physical network is plan of the network with real hardware components and its establishing.

5 Conclusion and outlook

In this paper we have developed the ARIS Unified Information System Development Process (AUP). The AUP arose from the Rational Unified Process (RUP) by extension of components which are defined in the ARIS framework concept. The ARIS framework concept considers besides the software development the construction of databases, networks, and its integration. In the ARIS framework concept an life cycle model is presented, which is similar to the well-known waterfall model in the software development. As the RUP is determined from the waterfall model by adding several iterations, the AUP is also determined from the ARIS life cycle by adding several iterations. The modeling language for the AUP is an aggregation of the UML models and the ARIS models.

The main benefit which we expect by using the AUP is to detect errors in the information system development earlier. This would reduce the costs of information system development enormously. We derive this assumption from the fact that RUP has reduced costs in pure software development projects. The main reason for earlier error detection is the close connection between customer and implementer.

This paper is an overview paper similar to [RUP] for the RUP. In the next time this paper is to advance to a full description of the AUP as it is done for example in [Kr03] for the RUP.

Furthermore we will build “AUP the product”. Analogous to RUP the product, it should offer some manuals and a configurable online version.

The modeling language is as said before an aggregation of the UML models and the ARIS models. The UML models are standardized in the look and feel. The ARIS models in contrast are only defined on the meta level by class diagrams. A further purpose is to define a unified language for the AUP, where the ARIS models satisfy the defined class diagrams on the meta level and the UML diagrams are given in version 2.0 [HJR+04].
References


