# Functional size measurement applied to UML-based user requirements

Klaas van den Berg, Ton Dekkers, Rogier Oudshoorn

## Abstract

*There is a growing interest in applying standardized methods for Functional Size Measurement (FSM) to Functional User Requirements (FUR) based on models in the Unified Modelling Language (UML). No consensus exists on this issue. We analyzed the demands that FSM places on FURs. We propose a requirements space with several levels of refinement, and show how UML can be used to specify FURs at these levels. FSM can be applied at the product level of UML-based FURs. We discuss our experience for three case studies and with two FSM methods: Function Point Analysis (FPA) and COSMIC-Full Function Points (CFFP).*

Keywords:
Functional Size Measurement (FSM), Function Point Analysis (FPA), COSMIC-Full Function Points (CFFP), Unified Modelling Language (UML), Functional User Requirement (FUR)

## 1. Introduction

There is a growing interest in applying standardized methods for Functional Size Measurement (FSM) to Functional User Requirements (FUR) based on models in the Unified Modelling Language (UML) [1]. Functional Size is defined as a size of software derived by quantifying the Functional User Requirements [8]. Functional User Requirements represent the user practices and procedures that the software must perform to fulfil the user's needs. The UML is a widely accepted language - in industry and academia - for specification and design of information systems [18].

There is no consensus on how to apply FSM to UML-based functional user requirements. The use of only some UML models has been investigated in literature. Also, multiple FSM methods are available. We have set out to research the applicability of two FSM methods, Function Point Analysis (FPA) and COSMIC-Full Function Points (CFFP), on the wide range of options in UML-based functional user requirements. Moreover, we wish to know which method is best suited for measuring UML-based FURs.

Our research builds on work of Fetcke [4], Bevo [1] and Jenner [9][10]. Fetcke [4] analyzed the applicability of FPA on a UML predecessor, and his mapping is mostly conformed by newer work [6]. Bevo [1] defined a mapping between use cases and COSMIC-FFP v1.0. Jenner [9][10], building on Bevo's work, used sequence diagrams for FSM. Jenner's proposal has been added to the COSMIC-FFP guideline [12].

This paper is structured as follows. In section 2 we discuss our approach and introduce an FSM process model, the two FSM methods and UML. In section 3 we analyze the demands FSM puts on FURs and the usage of UML in specifying FURs. In section 4 we introduce the case studies. The measurement results are discussed in section 5. In section 6 we discuss related literature, and in section 7 we present conclusions and recommendations.

## 2. Approach

In order to investigate the applicability of FSM to UML-based FURS, we set up our research into three parts:

1. In the analytical part, we create a requirements space and made conjectures on the applicability of FSM on UML-based FURs at various levels of refinement.
2. In the empirical part, we check our conjectures in three case studies.

3. In the conclusive part, we compare measurements of UML-based FURs with the two FSM methods FPA and COSMIC-FFP.

In the analytical part, we first analyze two FSM methods. Based on this analysis, we specify a number of demands these methods place on FURs to enable proper measurement. We also analyze UML's ability to represent FURs at different levels of refinement. This requirements space combined with the demands from FSM, gives a view of at what level of refinement FSM can be applied to FURs.

In the empirical part, we compare FSM results of a UML-based FUR to a traditional (non UML) view of the same FUR as outlined in standard ISO 14143-4 [8]. We adapt their benchmark process model, intended to compare FSM methods (see Figure 1), to compare different representations of the same FUR. ISO 14143-4 also provides case studies. We use the Hotel Case as our main case study. We use two other cases for additional experience: the Library Case [16] and the Security Case [13] used as well by Bevo and Jenner.

The final part is based on both the actual measurements as well as our experience in applying the FSM methods.

## 2.1. Process Model

We adapted the process model of ISO 14143-4 [8] (see Figure 1) and identified the Reference FUR (2.1) and the UML-based FUR (2.2). For three case studies we transformed the Reference FURs into UML-based FURs, which enables comparison of measurements with FSM methods.

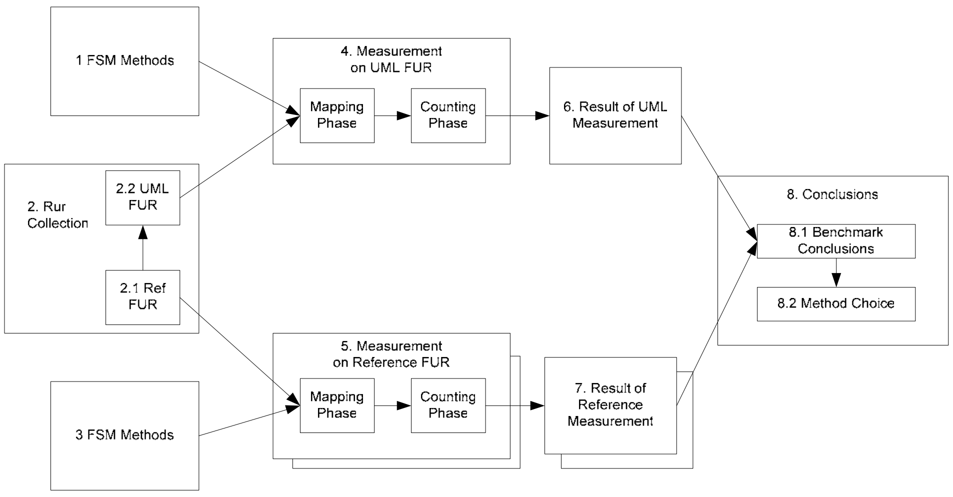

*Figure 1 - Process Model, based on ISO [8]*

Four ISO-certified Functional Size Measurement methods currently exist. These are IFPUG Function Point Analysis [4], Mark II Function Point Analysis [17], NESMA Function Point Analysis [14] and COSMIC Full Function Points [2]. The NESMA and IFPUG methods are alike and the Mark II method is used less frequently. For this paper, we analyzed the NESMA-FPA method and the COSMIC-FFP method.

The ISO-certified FSM methods have a user-view of software; they measure the functionality of the system which is visible for its users. The methods consist of two phases (see Figure 1, in boxes 4 or 5). In the mapping phase the FUR is mapped into a model with so-called Base Functional Components (BFCs). In the counting phase this model is quantified by grading these BFCs.

## 2.2. NESMA-FPA

In NESMA-FPA [14], the BFCs are transactions. Each transaction (shown in Figure 2 as either an arrow for a functional transaction or an oval for a logical transaction) has a certain type and functionality:

- **External Inputs (EI).** This functional transaction moves data into the application without performing data manipulation.
- **External Outputs (EO).** This functional transaction moves data towards the user, and performs some data manipulation.
- **External Inquiries (EQ).** This functional transaction moves data towards the user, and does not perform data manipulation.
- **Internal Logical Files (ILF).** This logical transaction is persistent data maintained by the application through the use of EIs.
- **External Interface Files (EIF).** This logical transaction is persistent data used by the application, but not maintained by it.

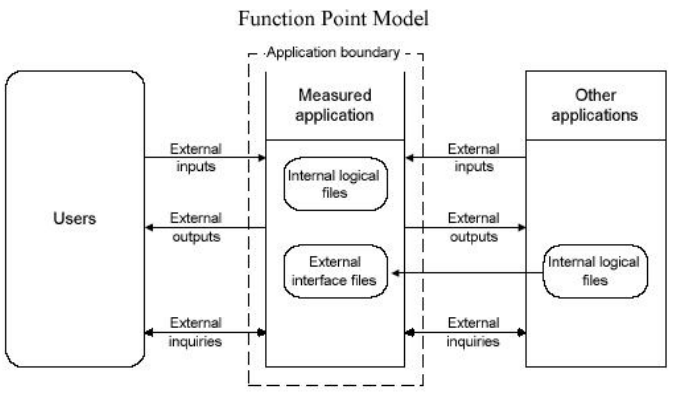

*Figure 2 - Function Point Model [4]*

In the counting phase, these transactions are graded by the amount of data they use. The logical transactions (or Files) are graded by the number of their entities (named Referenced Entity Types - RETs) and attributes (named Data Entity Types - DETs). The functional transactions are graded by the number of attributes (DETs) moved over the boundary and the number of referenced logical transactions. Then they are graded to be low, average or high and are assigned a number of Function Points (FP) accordingly.

## 2.3. COSMIC-FFP

In COSMIC-FFP [2], the BFCs are data movements, shown as the open arrows in Figure 3.

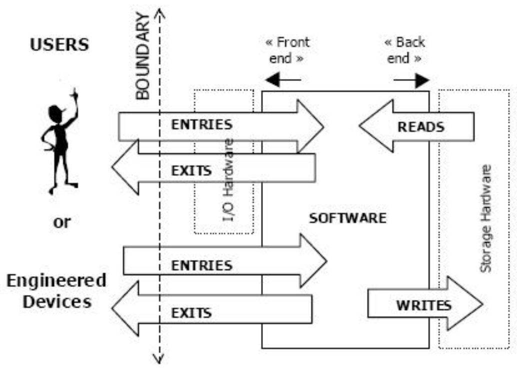

*Figure 3 - COSMIC-FFP Model [3]*

The data movements are grouped into Functional Processes, being a user-triggered function to perform a certain user-identifiable goal (a Functional Process is at the same level of abstraction as an FPA Functional Transaction). In a Functional Process, data is moved between user and system, or between system and persistent data. A Data Movement is the movement of a Data Group, which are subsets of entities.
The following data movements exist:

- **Entry**, from user to system
- **Exit**, from system to user
- **Read**, from persistent data to system
- **Write**, from system to persistent data

In the counting phase, each Data Movement is graded as 1 COSMIC Functional Size Unit (Cfsu) and all grades are added.

## 2.4. The Unified Modelling Language

UML is *a language for visualizing, specifying, constructing, and documenting the artefacts of a software-intensive system*. These artefacts are models represented in diagrams. UML 2.0 [18] defines a wide variety of diagrams (see Figure 4).

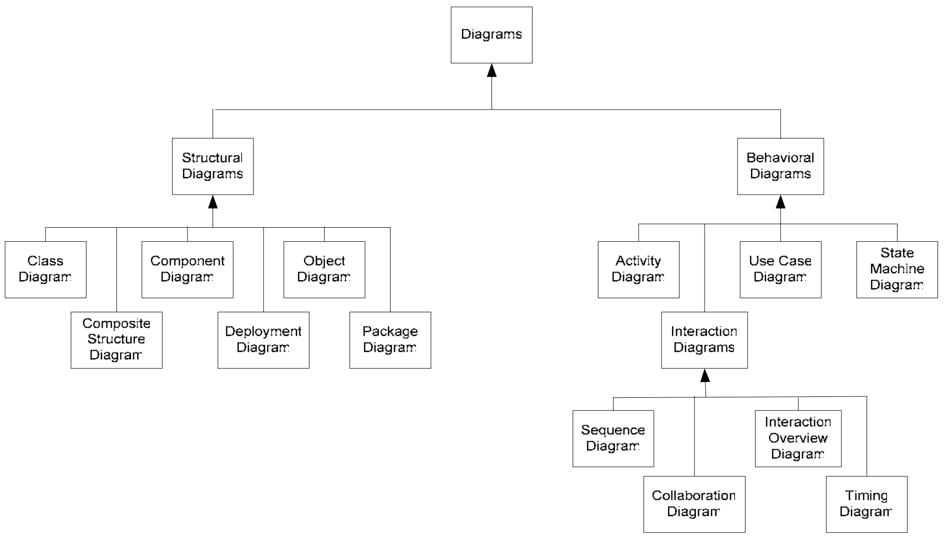

*Figure 4 – UML Diagrams 2.0 [18]*

The UML has two main types of diagrams, structural and behavioural. An interaction diagram is a subtype of behavioural diagrams, showing behaviour in combination with structural elements. We only briefly discuss diagrams used in this paper.

**Use Case Diagram**: *A diagram that shows the relationships among actors and the subject (system), and use cases.* [18]. This diagram visualizes the system as a box and use cases as ovals. A use case represents a function of the system, with actors outside the box accessing it. Use cases can be described at various level of formality. These descriptions usually include pre-conditions and post-conditions, as well as a step-by-step description of events (the flow of events). A path through a use case is known as a scenario.

**Activity Diagram**: *A diagram that depicts behaviour using a control and data-flow model.* [18]. This diagram can have an integrated perspective (just showing what is done, abstracting from who does what), but it can also use swimlanes to assign behaviour to actors or systems.

**Class Diagram**: *A diagram that shows a collection of declarative (static) model elements, such as classes, their content and relationships.* [18]. This diagram is used to model classes, their attributes and relations. It can be used to model other elements such as operations, but that is out of scope of this paper.

## 3. UML-based Functional User Requirements

First we analyze the FSM methods FPA and COSMIC-FFP in order to determine their demands on requirements in general. Next using Lauesen's [11] levels of requirements, we create a requirements space to show possible usage of UML models and diagrams in user requirements. Then we place the FSM demands in a UML perspective and select a set of UML diagrams for our case studies.

### 3.1. Required properties of Functional User Requirements

In order to carry out the mapping phase for FSM methods, a FUR must have certain properties [14]. By tracing the measurement process of both methods, we derived 6 demands shown in Table 1.

*Table 1- FSM required properties of FURs*

| Demand | General Description of Required Property | Required for FPA: | Required for CFFP: |
|---|---|---|---|
| 1 | A model of the logical data collection | Logical Transactions | Data Groups |
| 2 | The record and data element types of the logical data collection | Grading Logical Transactions | N.A. |
| 3 | System and Users have to be defined | Application Boundary | Application Boundary |
| 4 | An indication if (parts of) the data collection is being maintained by this or another system | Assessment of Logical Transactions; ILF or EIF | N.A. |
| 5 | A model showing the system functions including their in- and outgoing flow of information and their references in the logical data collection, including support functions (such as help files). | Locating Functional Transactions and grading them | Locating Functional Processes and their data movements |
| 6 | A detailed description of the in- and outgoing flows of information to the level of data element types. | Grading of Functional Transactions | Assessment of the Data Movements |

### 3.2. UML Requirements Space for FSM

According to Lauesen [11], user requirements can be specified at four levels of refinement:

1. **Goal-level requirements**. Specifying the business goal of a system.
2. **Domain-level requirements**. Specifying the domain in which the system is going to achieve its goal by describing high-level functions and its support.
3. **Product-level requirements**. Specifying the product by its inputs and outputs.
4. **Design-level requirements**. Specifying the product in exact detail.

The Unified Modelling Language can express these requirements on levels 2, 3 and 4. Goals at level 1 can not be expressed in UML models. We briefly discuss the UML diagrams (see Figure 4) usable in these levels. We propose a requirements space as shown in Table 2.

When describing the domain level one could define the context and the functional domains of a system by using a use case diagram. A high-level class diagram could describe the domains of data to be used by the system.

At the product level the domain level use case diagram could be refined, adding some functional decomposition and ensuring that the diagram covers the entire system. At this point, use case descriptions should be added to describe the functionality. UML offers various behavioural models to visualize use case descriptions. Interaction models can also be used by abstracting from the structure by modelling that structure as a single system entity. Class diagram could be used to model data and structural relationships (such as traditionally in Entity Relation Diagrams).

At design level we should again refine the product level artefacts. Instead of refining use cases, we should focus on describing scenarios. Scenarios can be described with a variety of behavioural and interaction diagrams much like at the product level. Class diagram may be refined further at this level.

*Table 2 – UML Requirements Space*

| ***Level*** | *Model Type* | *Diagrams* | *Describing* |
|---|---|---|---|
| **Goal** | None | N.A. | N.A. |
| **Domain** | Behavioural | Use Case Diagram | Actors, System, Global Functions |
| | Structural | Class Diagram | Global Information needs |
| **Product** | Behavioural | Use Case Diagram | Actors, System, Functions |
| | | Use Case Description | Events inside Use Case |
| | | Activity Diagram | Events inside Use Case |
| | | Interaction Overview Diagram | Events inside Use Case |
| | | State Machine Diagram | Events inside Use Case |
| | | Sequence Diagram | Events inside Use Case |
| | | Collaboration Diagram | Events inside Use Case |
| | Structural | Class Diagram | Data needed to support the Use Cases |
| **Design** | Behavioural | Use Case Diagram | Actors, System, Functions |
| | | Use Case Description | Events inside Use Case |
| | | Activity Diagram | Scenario, showing activities and exchanged data |
| | | Interaction Overview Diagram | Scenario, showing activities, exchanged data and control flow |
| | | Sequence Diagram | Scenario, showing exchanged data |
| | | Collaboration Diagram | Scenario, showing exchanged data |
| | Structural | Class Diagram | Data needed to support the Use Cases |

## 3.3. Applicability of FSM at levels of requirements space

When considering the FSM demands from section 3.1, FSM should be applicable at the product level:

- The persistent data, its attributes and relations are identified at that level.
- The context of actors and system is identified at the domain level, and so should the maintenance context be.
- The behavioural description should be detailed enough at the product level. The main focus of FSM, measuring the data which flows from and to the user, should be specified here.

This leads to the conclusion that we should use a set of UML diagrams to represent FURs at product level.

Considering demands 1and 2 from section 3.1, we require class diagrams to measure data structure and used attributes. Considering demands 3 and 4 from section 3.1, we require the use of use case diagrams in which the actors and system boundary are defined. Considering demands 5 and 6 from section 3.1, we require the use of use case descriptions: the functionality is explained in natural or some semi-formal language. This approach is dangerous because of the granularity pitfall [4][6][9][10]; we should be aware of ‘hidden’ functionality. Therefore, the addition of a behavioural diagram as used in [15] would be useful. By modelling the flow of events, it will be easier to locate functional transactions and functional processes in use cases.

For behavioural modelling of requirements, we choose the use of activity diagrams but other choices are feasible.

## 4. Case Studies

We investigated three cases (see Table 3): the Library Case [16] used by Sogeti in their FSM training, the Security Case [13] used by [1] [9] and the Hotel Case [8].

We provide an overview of the Hotel Case and use this as an example to discuss the refinement levels of UML for specifying FURs. The other cases will not be presented here due to lack of space.

*Table 3 - Case Studies*

| *Case* | *Size in Cfsu* | *Size in FP* | *Pages of FUR* | *Expressed in:* |
|---|---|---|---|---|
| **Library** | 101 | 101 | 10 | Traditional (non UML), screen shots |
| **Security** | 142 | 153 | 15 | UML, Sequence & Use Case |
| **Hotel** | 57 | 76 | 11 | Traditional (non UML), screen shots |

### 4.1. The Hotel Case

The Hotel Case is a relatively small case study, containing however a number of pitfalls and some odd design decisions. As such, it is an excellent case to use for a thorough empirical analysis. It comprises the creation and updating of reservations. We consider a Reservation Subsystem of the larger Hotel System. As shown in Figure 5, it has two main use cases: create Reservation and Change Reservation using the secondary use case Confirm Reservation.

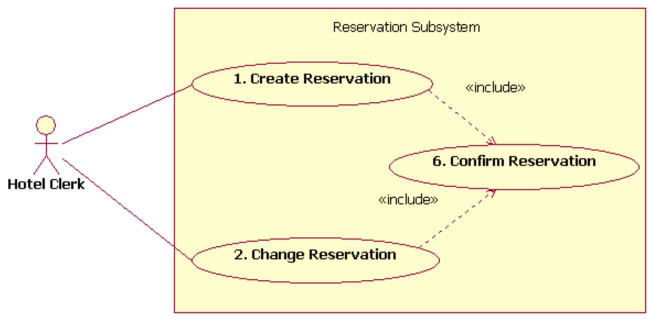

*Figure 5 – Product level Use Case Diagram of the Hotel Case*

### 4.2. Functional User Requirements

Setting up the UML-FUR is part of step 2 in the process model (box 2 of Figure 1). As an example, we highlight the use case Create Reservation by discussing its activity diagram (see *Figure 6*).

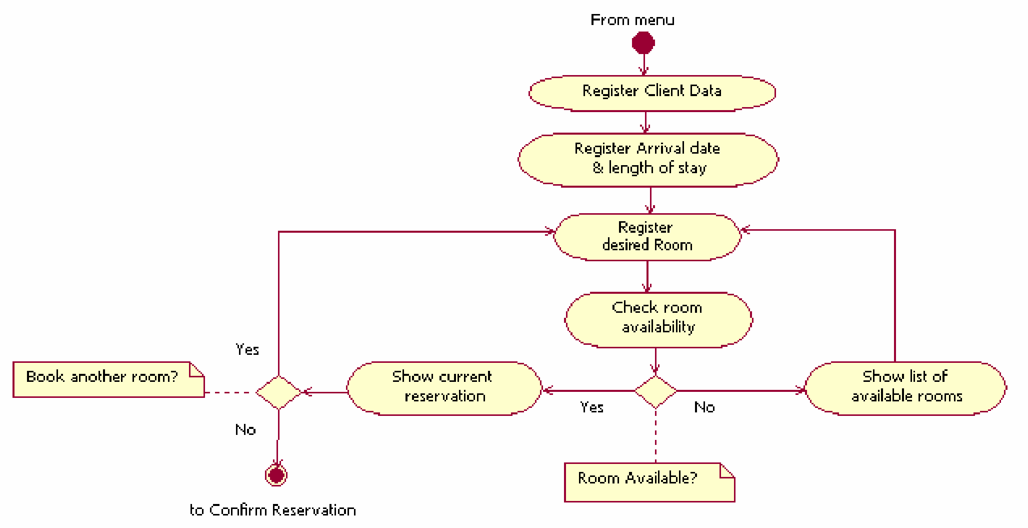

*Figure 6 – Activity Diagram of Use Case 1. Create Reservation*

This diagram shows activities undertaken in creating a reservation; first the client is identified and then the desired stay and room are selected. If the room is available, an overview of the reservation is given. If the room is unavailable, an overview is given of the available rooms. From there, the room selection can be changed. After the overview, a new room can be added in the same way, or the reservation can be confirmed.
Persistent data in the system is modelled in a class diagram as shown in Figure 7.

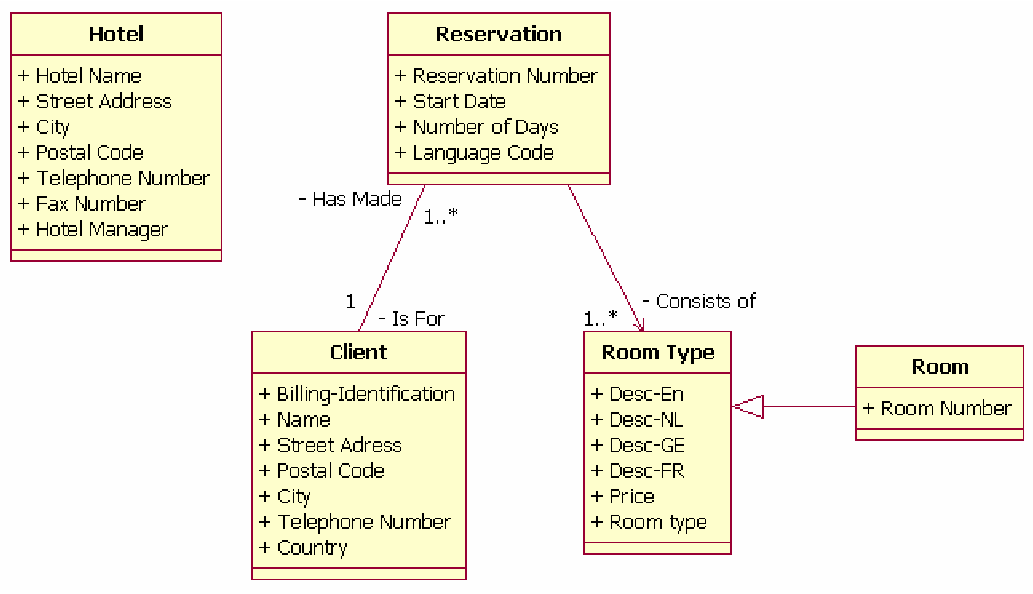

*Figure 7 – Class Diagram for Hotel Case*

## 4.3. Mapping Phase

The mapping phase is the first step in the measurement phase of the process model (box 4 of Figure 1). For FPA in this Hotel Case, multiple transactions can be identified. The main user-identifiable goal is to enter a reservation, an External Input. From the activity diagram it can be seen that this comprises of four activities. The show activities serve different user-identifiable goals; one is to give an overview of the current reservation and one to show the free rooms. These are two external outputs.

Under the COSMIC-FFP guideline for sizing business application software, previewed by [12], a functional process must be independently executable and triggered in the world of users. When in doubt, any user decision triggers a new process. This example is triggered to book a desired room; this is done after the third activity. The fourth activity is triggered by the user to check if this is possible. However, this is no real decision; it is implied that when you book a room you want to see if it is feasible. Hence, the functional process ends either with an overview of the reservation or with the overview of the available rooms. Selecting a new room is a decision, but it leads back to the already identified process. Adding a new room is a decision, which also leads back into the already defined process. The alternative to adding a new room, the end point in the Activity Diagram, leads to another use case to confirm the reservation. As confirming the reservation is not the reason to trigger this functional process (it is triggered to make a reservation), Confirm Reservation is another functional process.

## 4.4. Counting Phase

The counting phase is the second step in the measurement phase of the process model (Figure 1, box 4).

*Table 4 – FPA count for Hotel Case*

| Transaction | Type | Referenced Files | Referenced Attributes | Extra DETs | FP |
|---|---|---|---|---|---|
| Register Reservation | EI | 3 | 14 | 2 | 6 |
| Show current reservation | EO | 3 | 5 | 2 | 5 |
| Show available rooms | EO | 3 | 7 | 2 | 5 |
| Total | | | | | 16 |

The FPA count (in FP's) is presented in Table 4. Per transaction, the referenced files and attributes are listed. The extra DETs are added per the guidelines [14] to account for the access via a menu structure and possible error messages.

*Table 5 – COSMIC-FFP count for Hotel Case*

| Activity | BFC | Number | Cfsu |
|---|---|---|---|
| Register Client Data | Entry | 1 | 1 |
| Register Arrival date & length of stay | Entry | 1 | 1 |
| Register Desired Room | Entry | 1 | 1 |
| Check Room availability | Read | 3 | 3 |
| | Write | 2 | 2 |
| Show available rooms | Exit | 2 | 2 |
| Show Current reservation | Exit | 3 | 3 |
| End of Functional Process | Exit | 1 | 1 |
| Total | | | 14 |

The COSMIC-FFP count (in Cfsu's) is presented in Table 5. Per activity the relevant data movements have been listed and graded to the number of data groups.

## 5. Discussion

This section addresses the phases Measurement and Conclusion from the process model (boxes 6, 7 and 8 from Figure 1).

We compare our previous results to results based on the traditional (non UML) FUR from [8]. These results were obtained by professionals at Sogeti Nederland. The results for the Hotel Case were very good; the measurement variances are never higher then five percent (which is a commonly accepted variance in practice). The variance that occurs is due to the variance in the database model versus the class diagram. The reference database model used code tables; an implementation aspect ignored in UML (and COSMIC-FFP) but not in NESMA FPA. Code tables should not be in FURs as they are technical aspects [7].

What is interesting in these measures is the ease and speed of applying FPA when compared to COSMIC-FFP. FPA has strict defined transactions which are easier to map on FURs. On the other hand COSMIC-FFP has functional processes. Mapping complex functions onto functional processes is nontrivial. Assessing a function in terms of input/output as in FPA provides more intuitive results then assessing its triggers as in CFFP, as shown in the Hotel Case.

For COSMIC-FFP, both phases Measurement and Conclusion can not be automated at product level of refinement of FURs. From our example, it is not clear from the activity diagram whether the choice between Show Reservation and Show Available Rooms is made by the system or by the actor. At the counting phase, the data movements are drawn from the use case descriptions and not from the activity diagram. The same applies for FPA. The activities in an activity diagram are not typed as being input or output. The counting phase has similar problems; the attributes moved and files referenced are not defined in the diagram but in the use case descriptions.

## 6. Related Work

Several papers in literature discuss FSM of FURs with UML diagrams at various levels using various FSM methods. We discuss some papers related with our research.

Fetcke [4] kicked off the issue in 1998 by creating rules and mapping steps to conform IFPUG FPA's (v4.0) mapping phase to Object Oriented Software Engineering (OOSE, an UML predecessor) modelling. He employed use cases to locate the context aspects and functional transactions. He used an OOSE-specific model for the logical transactions, but a class diagram could easily substitute. His mapping is regarded as being largely correct by later literature [6]. Apparently, he measured at product level, which supports our analysis.

Bevo [1] measured similar to Fetcke, but based on COSMIC -FFP v1.0. He questioned if use cases or scenarios were the best candidates as Functional Process equivalents. He concluded that counting based on scenarios could give a much larger functional size then using just use cases.

In hindsight it has been concluded by Jenner [9] that Bevo did not compare use cases to scenarios but rather used two levels of use cases. In terms of our analysis, he used domain level use cases (which he called use cases) and product level use cases (his scenarios).
Also, Bevo's counts were made using another interpretation of the COSMIC-FFP counting and mapping rules then are used today. As such, his numerical results can not be evaluated easily.

Jenner built upon Bevo's work and defined the granularity issue; the fact that use cases can have different levels of refinement. Jenner proposed using another UML diagram, the sequence diagram, as the primary diagram to count COSMIC-FFPs. Jenner then counted the same case as Bevo with his method, and found an even higher result by not using one but multiple functional processes per scenario. Apparently, Jenner measured at design level as he used sequence diagrams to visualize single scenarios. His proposal of sequence diagram usage is published in the COSMIC-FFP Guideline. Jenner though, also used a different interpretation of the COSMIC-FFP method making a comparison to his and Bevo's work hard.

In the COSMIC Guideline for sizing business application software, previewed by [12], references are made to the UML. Here, mapping rules are given to handle inheritance issues for sizing. Also, the Guideline stresses that both behavioural and structural information is needed for accurate sizing. The Guideline sees use cases as something completely different as functional processes, as they may be any number of use cases covering any number of functional processes. In the appendix to this Guideline, measurement based on sequence diagrams is described as very powerful and explained further (in line with Jenner). This Guideline provides an accurate coverage on how to handle design level requirements, as described in this paper.

## 7. Conclusion

We first identified required properties of FURs to enable FSM with FPA and COSMIC-FFP. Then, we proposed a requirements space for UML-based requirements in four refinement levels. Our analysis indicated that FSM on requirements expressed in UML is possible from the product level, as well for FPA as for COSMIC-FPP. At further refinement, the design level, FSM is supported in the official CFFP Guideline. We checked our analysis in three cases. Our results showed that our analysis was correct as the results were on the mark for both methods. Differences remain between FPA and COSMIC-FPP, but these are not related to the use of UML in FURs. There was a difference in mapping, due to a non-functional requirement in the reference FUR measured only in FPA and not in COSMIC-FPP. The FPA method however, is easier to use. This method has stricter rules on its behavioural measure. More research is needed on the proposed UML-based requirements space. A better view on what models should be used on different levels of refinement would improve the applicability of FSM methods to UML-based FURs.

## References

[1] Bevo, V., Levesque, G. & Abran, A. (1999). “Application of the FFP method to a specification in UML notation: account of the first attempts at application and questions arising.” Translated by M. Jenner. French original ‘Application de la méthode FFP à partir d'une spécification selon la notation UML: compte rendu des premiers essais d'application et question’ retrieved May 17, 2004 from http://www.lrgl.uqam.ca/publications/

[2] Booch, G., Rumbaugh, J. & Jacobsen, I. (1999). “The Unified Modeling Language User Guide”. Addison Wesley Longman, Massachusetts, USA.

[3] COSMIC (2003). “COSMIC FFP Measurement Manual (version 2.2)” Retrieved march 20, 2004, from http://www.cosmicon.com/

[4] Fetcke, T., Abran, A. & Nguyen, T. (1998). “Mapping the OO-Jacobsen Approach to Function Points.” Proceedings of Tools 23’97 – Technology of Object Oriented Languages and Systems. IEEE Computer Society Press; Los Calamos, California, United States.

[5] IFPUG (1999). “Function Point Counting Practices Manual (Release 4.1).” Westerville, USA: IFPUG

[6] Iorio, T. (2004). “IFPUG Function Point Analysis in a UML framework.” Proceedings of SMEF 2004; Rome, Italy.

[7] ISO (1998). “ISO/IEC DTR 14143-1 (Definition of Concepts)”. Retrievable from www.iso.org

[8] ISO (2003). “ISO/IEC DTR 14143-4 (Reference Model)”. Retrievable from www.iso.org

[9] Jenner, M. (2001). “COSMIC FFP 2.0 and UML: Estimation of the size of a system specified in UML – Problems of granularity.” Proceedings of FESMA-DESMA Conference 2001, Heidelberg, Germany.

[10] Jenner, M. (2002). “Automation of Counting of Functional Size using COSMIC-FFP in UML”. Proceedings of 12th International workshop on Software Measurement, IWSM 2002, Magdeburg, Germany.

[11] Lauesen, S. (2002). “Software Requirements – Styles and Techniques.” Addison-Wesley, London, UK.

[12] Lesterhuis, A.. & Vogelezang, F.W. (2004) “Guideline for the application of COSMIC-FFP for sizing Business Applications Software.”, previewing the guideline by Lesterhuis, A. & Symons, C. (2004). Proceedings of the International workshop on Software Metrics, IWSM 2004, Magdeburg, Germany.

[13] Muller, P. (1997). “Instant UML.” Wrox Press Ltd, Birmingham, UK.

[14] NESMA (2004). “Definities en telrichtlijnen voor de toepassing van Functiepuntanalyse.” (in Dutch, Versie 2.2). Zeist, the Netherlands: NESMA.

[15] Schneider, G. & Winters, J.P. (1998). “Applying Use Cases – A Practical Guide.” Reading, USA: Addison-Wesley.

[16] Sogeti (2003). “Library Case v2.3” Den Bosch, the Netherlands: Sogeti Nederland B.V.

[17] Symons, C.R. (1991). “Software Sizing and Estimating – Mk II FPA.” Chichester, England: John Wiley & Sons Ltd

[18] UML (2003). “UML 2.0 Superstructure (draft)”. Retrievable from ww.omg.org