

Translating RAD Business Process Models into BPMN Models: A Semi-Formal Approach

Rana Yousef¹, Mohammad Odeh¹, David Coward¹, Ahmad Sharieh²

¹Bristol Institute of Technology

The University of the West of England, Bristol, UK

²King Abdullah II School for Information Technology

The University of Jordan, Amman, Jordan

{rana2.yousef, mohammed.odeh, david.coward}@uwe.ac.uk, sharieh@ju.edu.jo



ABSTRACT: *Modelling business processes using BPMN has been emerging as the preferred choice for organisations seeking to enact their business processes in service-oriented environments. However, a number of organisations have their business processes modelled using Role Activity Diagramming, RAD. In this research, we introduce a new approach to translating RAD business process models to their respective BPMN ones carried out in two stages starting from identifying the corresponding BPMN notation to RAD modelling constructs, and then introducing a new algorithm to translate a given RAD business process model to its suggested corresponding BPMN model demonstrated by an example from the healthcare domain. This translation revealed that all RAD process elements can be mapped to BPMN. However, there still remains space for the business process modeller to enhance the newly generated BPMN model with the extra rich features of BPMN. Finally, this work makes a further contribution to the emerging trend of bridging the gap between business process models and systems in addition to paving the way for the migration of legacy RAD models to BPMN for their enactment in SOA environments.*

Keywords: RAD, BPMN, Business Process Model

Received: 11 June 2011, Revised 9 August 2011, Accepted 17 August 2011

© 2011 DLINE. All rights reserved

1. Introduction

Business Process Modelling Notation (BPMN) is the emerging standard for modelling business processes [4]. Its underlying mapping to executable process languages such as BPEL makes it suitable for aligning business needs and IT capabilities [3]. Thus, this motivates considering BPMN an important standard for SOA-based systems [3].

It is accordingly desirable for organisations whose business processes are modelled using other notations, such as RAD, to translate their models into BPMN, especially if they are considering SOA. One such example in the healthcare domain is the modelling of the Cancer Care and Registration (CCR) process in Jordan [2], where a complete model of the process is available in RAD. Thus, translating these models into the respective BPMN ones is a first step towards enacting these non-BPMN models in SOA environments.

In this paper, we provide a novel algorithm to fully translate RAD process models to the corresponding BPMN ones. The CCR process [2] has been utilised as a case study to demonstrate this algorithm and also validate the newly obtained corresponding BPMN model.

The rest of the paper is organised as follows. Section 2 provides a brief overview of both RAD and BPMN, Section 3 explains

why it is desirable to have an algorithm to translate RAD models into BPMN, Section 4 provides a mapping between the key elements and process concepts used in the two modelling notations, Section 5 introduces the RAD to BPMN translation algorithm, section 6 applies it using an example from the healthcare domain and finally section 7 is the concluding section.

2. A Brief Overview of RAD and BPMN

2.1 RAD

Role Activity Diagramming provides a simple and intuitive means of visually representing business process models, which play a substantial role in understanding and improving the respective business processes. The basic concepts of RAD were enriched by Ould [6], where roles and the interactions between them form the basis of such models. A role can be thought of as a set of related activities that can be carried out by a machine, a person, or a group of people [1]. Accordingly, a process modelled using RAD depicts process roles, their associated components including activities, states, events, etc, and the interactions between the roles of a given process. Activities denote the items of work, states represent the order of activities and events are required before actions can be performed.

2.2 BPMN

BPMN is a rich process modelling notation that can be effectively used to model business processes understandable at all levels, from business users, business analysts, and process owners, to the technical architects and developers [3, 4]. BPMN has been developed under the auspices of the OMG (Object Management Group) and defines four categories of elements [3]: flow objects, connecting objects, swim lanes and artefacts.

Flow objects in BPMN are comprised of activities (tasks or sub processes), events (triggers or results) and gateways (to control sequential flows). Connecting objects are used to connect flow objects. The three types of connectors are: sequence flows, message flows and associations. Swim lanes are used to organise activities and pools. Finally, artefacts are used to include additional information annotations and data objects.

3. The Importance of the Translation Algorithm

We mentioned previously that it is desirable for organisations whose business processes are modelled using notations other than BPMN, such as RAD, to translate their models into BPMN, especially if they are considering SOA, because BPMN is becoming an important standard for SOA-based systems due to the underlying mapping of BPMN models to executable process languages such as BPEL.

It is accordingly worthwhile to perform the translation between the modeling notations in a systematic way to ensure a consistent and accurate translation process. This consistency is especially required when translating a large number of business processes for a whole enterprise. In addition, using a systematic way to translate process models provides better understanding of the way the translation is performed and reduces effort and duplicated tasks.






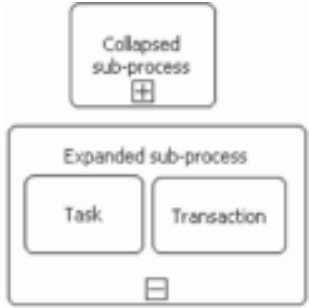
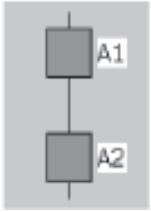


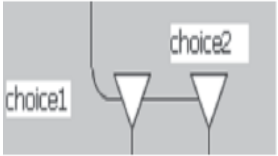


In this paper, we provide a novel algorithm that fully translates RAD process models to the corresponding BPMN ones in a systematic way.







4. Mapping RAD Modelling Constructs to BPMN Notation

Before we introduce our new algorithm to translate RAD business process models into the BPMN respective ones, we introduce in Table 1 and Table 2 a suggested mapping of RAD modelling constructs to the respective best matching BPMN notation. BPMN appears to be more expressive than RAD and thus extra features in relation to the same process concept in RAD briefly explained in the fourth column of Table 1. Tables 1 and 2 have been constructed based on information available in [1, 3, 4, 5].

5. Translating RAD Business Process Models to BPMN

In this section we present a new algorithm to translate a given RAD business process model to its corresponding BPMN model, the algorithm takes the RAD BP as input, makes visual interpretation of the model and then generates the respective BPMN model:

Key elements and process concepts	RAD	BPMN	Extra features in BPMN
<p>Role: An organising unit for binding activities that are strongly related</p>	<p>Role: a set of activities which when taken together achieve a goal</p> 	<p>Pool: represents a participant in a process</p> 	<p>Lane: used to better organise and structure pools</p> 
<p>Activity: A generic type of work that an organisation performs.</p>	<p>Activity or Action: the items of work that people do, represented as dark boxes within a role.</p> 	<p>Task: used to represent the activity on the lowest abstraction level.</p> 	<p>Compound processes: a set of tasks that can be collapsed into a sub-process.</p> 
<p>Ordering: The order of activities execution in a process.</p>	<p>A line connecting activities denotes a state or sub-state of a role. These states represent the order of activities.</p> 	<p>Sequence flow is used to show the order in which the activities in a process will be performed. The normal sequence flow is denoted as:</p> 	<p>There are other types of sequence flows in BPMN:</p> <ul style="list-style-type: none"> - Conditional sequence flow: the flow is associated with condition expressions evaluated at run-time to determine whether the flow will be used or not. - Default sequence flow: used with XOR (data) and OR to indicate that this flow is taken only if all other outgoing conditional flows are not true at runtime. 
<p>Choices: The conditions under which different activities take place</p>	<p>Case refinement: the conditions under which different activities take place.</p> 	<p>XOR (Data) Gateway: used for data-based exclusive decision or merging.</p> 	<p>Other types of choice gateways:</p> <p>XOR (Event): event based exclusive decision only.</p> <p>OR: Data based inclusive decision or merging.</p> <p>COMPLEX: Complex Condition (a combination of basic conditions).</p> 

<p>External Event: external events or inputs which are needed before work can continue</p>	<p>External events or triggers which are required before work can continue.</p> 	<p>Start and Intermediate events: in general the start event indicates where a process will start and intermediate events occur between start and end events. It will affect the flow of the process.</p> 	<p>Start and intermediate events in BPMN can be specified as follows:</p> <ul style="list-style-type: none"> - Timer Start/Intermediate Event: a specific time can be set to trigger the start of the process, or continue the process. - Message Start/Intermediate Event: a message arrives from a participant and triggers the event. - Rule Start/Intermediate Event: this type of event is triggered when the conditions for the rule become true. - Link Start/Intermediate Event: used to connect the end of one process to the start of another. - Multiple Start/Intermediate Event: indicates that there are multiple ways to trigger the end of the process. - Error Intermediate Event: indicates the generation of an error. - Compensation Intermediate Event: indicates the requirement for compensation, i.e. rolling back. 
<p>Goal:</p>	<p>State Marker: often used in RAD at the end of a process to indicate a goal satisfaction.</p> 	<p>End Event: in general the end event indicates the halt of the business process.</p> 	<p>End events can be specified as follows:</p> <ul style="list-style-type: none"> - Message End Event: indicates that a message is sent to a participant at the conclusion of the process. - Error End Event: indicates an error generation. - Cancel End Event: used with transaction sub-process. - Link End Event: used to connect the end of a process to the beginning of another. - Multiple End Event: indicates that there are multiple ways to trigger the end of the process. - Termination End Event: indicates that all activities in the process should be terminated. - Compensation End Event: emphasises the requirement for compensation, i.e. rolling back. 

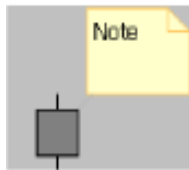

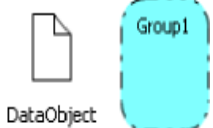
<p>Artefacts: Used to provide additional information about the process</p>	<p>Note: provide additional information for the reader of a RAD diagram.</p> 	<p>Annotation: text annotations can be used by the modeller to provide additional information for the reader of a BPMN diagram.</p> 	<p>Other types of artefacts are:</p> <ul style="list-style-type: none"> - Data objects provide information about what activities are required to be triggered and/or what they produce. Data objects do not have any direct effect on the sequence flow or message flow of the process. - Group: grouping can be used for documentation or analysis purposes; grouping does not affect sequence or message flows. 
---	--	---	---

Table 1. A suggested mapping of RAD modelling constructs to the respective best matching BPMN notation (part 1)


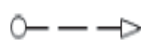
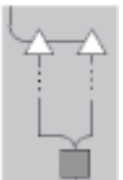





Key elements and process concepts	RAD	BPMN
<p>Interactions:</p>	<p>Interaction between roles: a line linking two interaction parts (white boxes) in different roles to denote synchronous interaction between them. A driving interaction part can be distinguished using diagonal lines inside it.</p> 	<p>Message flow: Used to show the flow of messages between two participants (represented as pools in BPMN)</p> 
<p>Concurrent threads:</p>	<p>Part refinement: shows activities in sub-threads to the main thread. Threads are joined when lines from multiples threads merge into one line.</p> 	<p>AND Gateway: used for parallel forking and joining (synchronization)</p> 
<p>Iteration</p>	<p>Looping is performed by returning to a previous state in the role. Another type of iteration in RAD is the replication of threads: A triangle and an asterisk also indicate that the activities below the triangle will be performed in parallel a number of times.</p> 	<p>Looping: can be performed by returning to a previous point in the process or using the loop marker within a task or a sub-process.</p> 
<p>Cardinality</p>	<p>Replicated pre-existing role: A role with a number, n, and a tick on top indicates that n instantiations of the role always exist.</p> 	<p>Multiple instances: multiple instances of a task or sub-process is created. A number, n, stored as an attribute value generally provided by the modeling tool indicates that n instances of that task or sub-process are created.</p> 

Table 2. A suggested mapping of RAD modelling constructs to the respective best matching BPMN notation (part 2)

Input: RAD business process model; a collection of roles $\{r_1, \dots, r_n\}$ where each role consists of elements $\{e_1, \dots, e_v\}$, where an element may be an activity, a case refinements, a part refinements, a state, a joining line of parallel threads, a trigger, an external event or a goal.

Output: A corresponding BPMN model; a collection of pools $\{p_1, \dots, p_j\}$ where each pool consists of 1 or more lanes containing the corresponding BPMN elements: a BPMN element could be a task, an XOR gateway, an AND gateway, a sequence flow or a start/Intermediate/End event.

Begin

Identify the set of all roles, R, in the RAD model: $R = \{r_1, r_2, \dots, r_n\}, 0 \leq i \leq n;$

For each role r_i in the RAD model do the following:

Map r_i to a pool $p_j, 0 \leq j \leq m;$

Mark the beginning of the process, $pr_k, 0 \leq k \leq s$ captured in r_i using the Start Event;

Identify the set of all elements, E, within $pr_k: E = \{e_1, e_2, \dots, e_v\}, 0 \leq l \leq v;$

For each elements e_l in pr_k map e_l as follows:

If e_l is an Activity or a driving interaction part or an equivalent interaction part then

Map e_l into a Task;

Else if e_l is a Case Refinement then

Map e_l to an XOR (date) gateway;

Else if e_l is a Part Refinement then

Map e_l to an AND gateway;

Else if e_l is a joining line of parallel threads then

Map e_l to an AND gateway;

Else if e_l is a trigger, external event or a goal then

Map e_l to the proper type of Start/Intermediate/End Event

End if;

End for each element;

Map all states to sequence flows;

Mark the end of the process pr_k using the End Event;

If r_i is a replicated pre-existing role then

Collapse elements into a sub-process marked with "Multiple instances"

End if;

End for each role;

Map interactions between roles to message flows between pools;

Map multiple non-driving interaction parts to XOR (event) gateway;

Map Notes in the RAD model to Annotations in the BPMN model to provide more information;

End

As can be seen from the above algorithm, each role is mapped into a pool. a pool can only contain one process, i.e. one start event and one end event, and of course message flows are not allowed within a process. In the case of replicated pre-existing role, the best act is to collapse the activities of that role into a sub-process marked with the "multiple instances" notation to indicate the multiple instances that should be created for that sub-process.

The different types of start, intermediate and end events can be used where appropriate to represent the triggers, external events and goals in the RAD model. In some cases, the presence of disconnected segments of activities within a role indicate that there are multiple ways to trigger the process within that role, in these cases the Multiple start/intermediate Events can be used. In other cases it is only done to enhance readability, in these cases Link Events may be used if it is difficult to connect the segments in a readable way.

Mapping elements within each role is done as follows: An activity can be mapped to a task with the possibility to gather some

tasks into a collapsed sub-process to increase modularity and/or readability of the diagram. Driving interaction parts and equivalent interaction parts can also be mapped to tasks as they can actually be thought of as actions, the non-driving interaction part, however, is not an action performed by the role but rather is an intermediate event that triggers an action. Case refinements are translated to XOR (Data) gateways. Nested case refinements in RAD can be simplified using other types of gateways, such as OR gateways and complex gateways. Part refinements are translated to AND gateways. This type of gateway is also used to join parallel threads when multiple lines in the RAD model merge into one line. States, which are used to order activities and connect objects in RAD models, are mapped to sequence flows in BPMN. Conditional sequence flows and default sequence flows can be used where appropriate. Interactions between roles are mapped to message flows between pools

Translating Loops is straightforward where in both models iteration is performed by returning to a previous point in the model. In BPMN a sub-process with a loop marker can be used either to represent a replication of threads or to make the model more readable.

Finally, artifacts that are used in a RAD model to provide additional information can be mapped into Annotations in the BPMN model. Data objects can also be used in BPMN models where the translator find appropriate to provide information about what activities are required and/or what they produce. flows can be used where appropriate. Interactions between roles are mapped to message flows between pools

Translating Loops is straightforward where in both models iteration is performed by returning to a previous point in the model. In BPMN a sub-process with a loop marker can be used either to represent a replication of threads or to make the model more readable.

Finally, artifacts that are used in a RAD model to provide additional information can be mapped into Annotations in the BPMN model. Data objects can also be used in BPMN models where the translator find appropriate to provide information about what activities are required and/or what they produce.

6. Example

In this section, we provide an illustration of the above algorithm, where it is applied to the process of Patient Reception of the overall CCR Process [2]. Figures 1 and 2 depict the patient reception process modelled in RAD and BPMN, respectively.

As can be seen from the figures, once the key elements of both modelling notations have been well understood to map to each other then the application of the above suggested RAD to BPMN translation algorithm is a straightforward process. In this example, the roles 'Receptionist, Patient and Medical Records' are mapped to the corresponding Pools, where each pool contains one process that begins with a start event and finishes with an end event. Activities, decisions, flows and interactions were translated as a one-to-one mapping according to the mappings presented in section 3. Events were inserted in the proper places in the BPMN model to keep the same semantics of that in the RAD model as was suggested in section 4, such as the Message Intermediate Events: request to make appointment and request to visit MR.

Let's take the Patient roles and give a step-by-step translation of its elements. The Patient role is mapped to a pool with the same name containing a process that begins with a start event and finishes with an end event.

The first activity in the role, visit clinic, is mapped to a corresponding task in the Patient pool. The two non-driving interaction parts within the role, which are considered as triggers for performing different actions, can be gathered using an Event-based decision. The first event is request to make appointment which triggers the task make appointment, the second event is the request to visit MR which triggers the task visit MR. After the latter activity there's an interaction part which receives a request to visit the receptionist and triggers a loop to the first activity in the role, visit clinic. This is mapped to an intermediate event receiving the request for visiting the receptionist and a line going back to the first task in the pool, visit clinic.

7. Conclusion

In this paper, we introduced a new approach to translating RAD business process models to their respective BPMN ones. This was carried out in two stages. First, identifying the corresponding BPMN notation to RAD modelling constructs in addition to identifying the richness of the similar BPMN compared to RADs. In the second stage, we introduced a new algorithm to

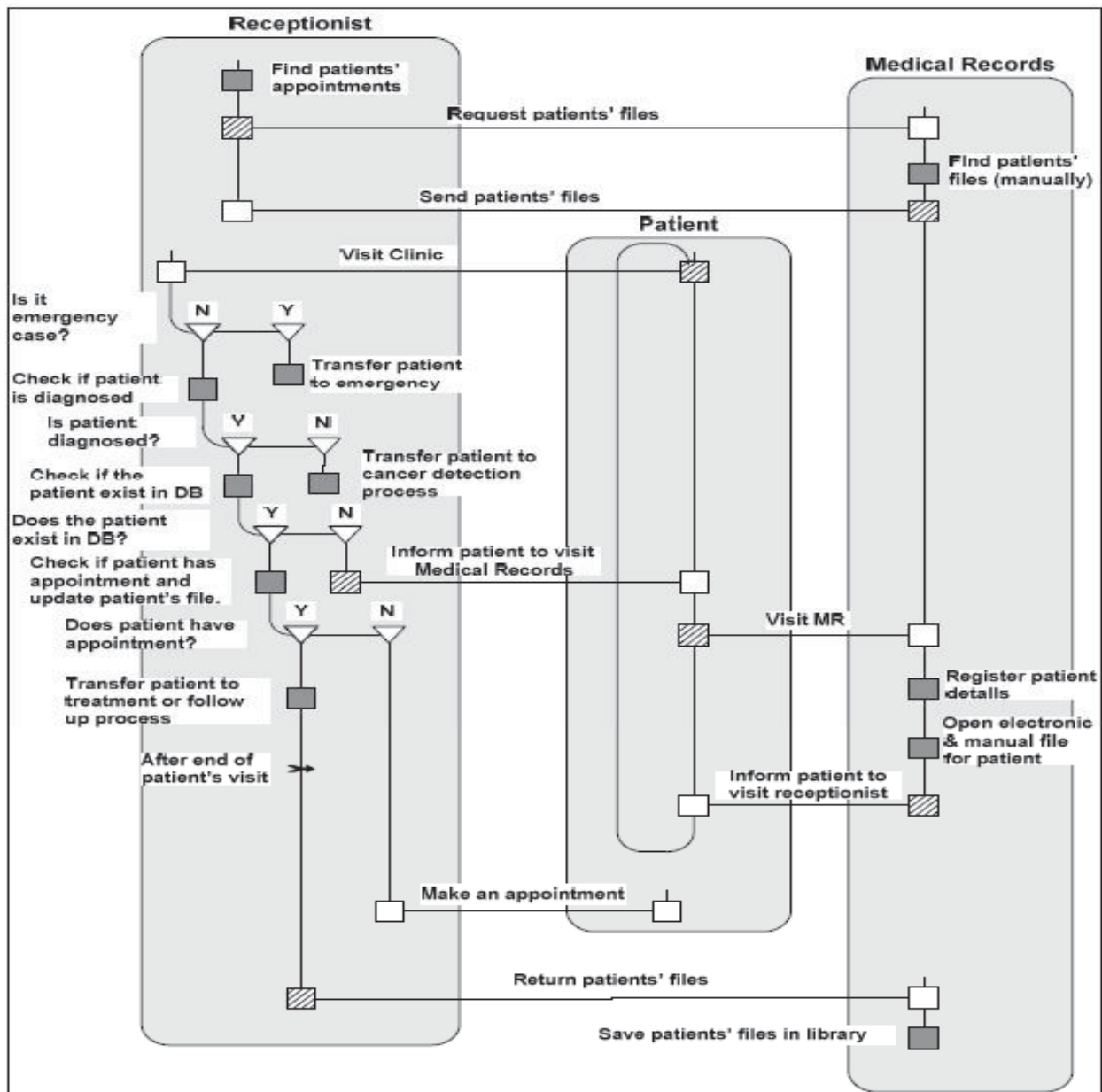


Figure 1. RAD model of the patient reception process

translating a given RAD business process model to its suggested corresponding BPMN model demonstrated by an example from the healthcare domain. This translation revealed that all RAD process elements can be mapped to BPMN. Although this translation process appears to be straightforward with a high degree of conformance of the generated BPMN model to the original RAD model, there still remains space for the business process modeller to enhance the newly generated BPMN model with the extra rich features of BPMN stemming from its expressiveness as an emerging business process modelling language. This, however, has strengthened our thinking behind undertaking this research, in the first place, so that not only RAD business process models are translated but that they can be further enriched and improved as part of the organisations' business process architecture. In addition, this makes a further contribution to the emerging trend of bridging the gap between business process models and systems, for example Odeh et. al [7] and also paves the way for the migration of legacy RAD models for their enactment in SOA environments due to the underlying mapping of BPMN models to executable process languages such as BPEL. Finally, it's worth mentioning that the introduced algorithm can be implemented where a computer program can be used to automatically convert RAD models into BPMN models.

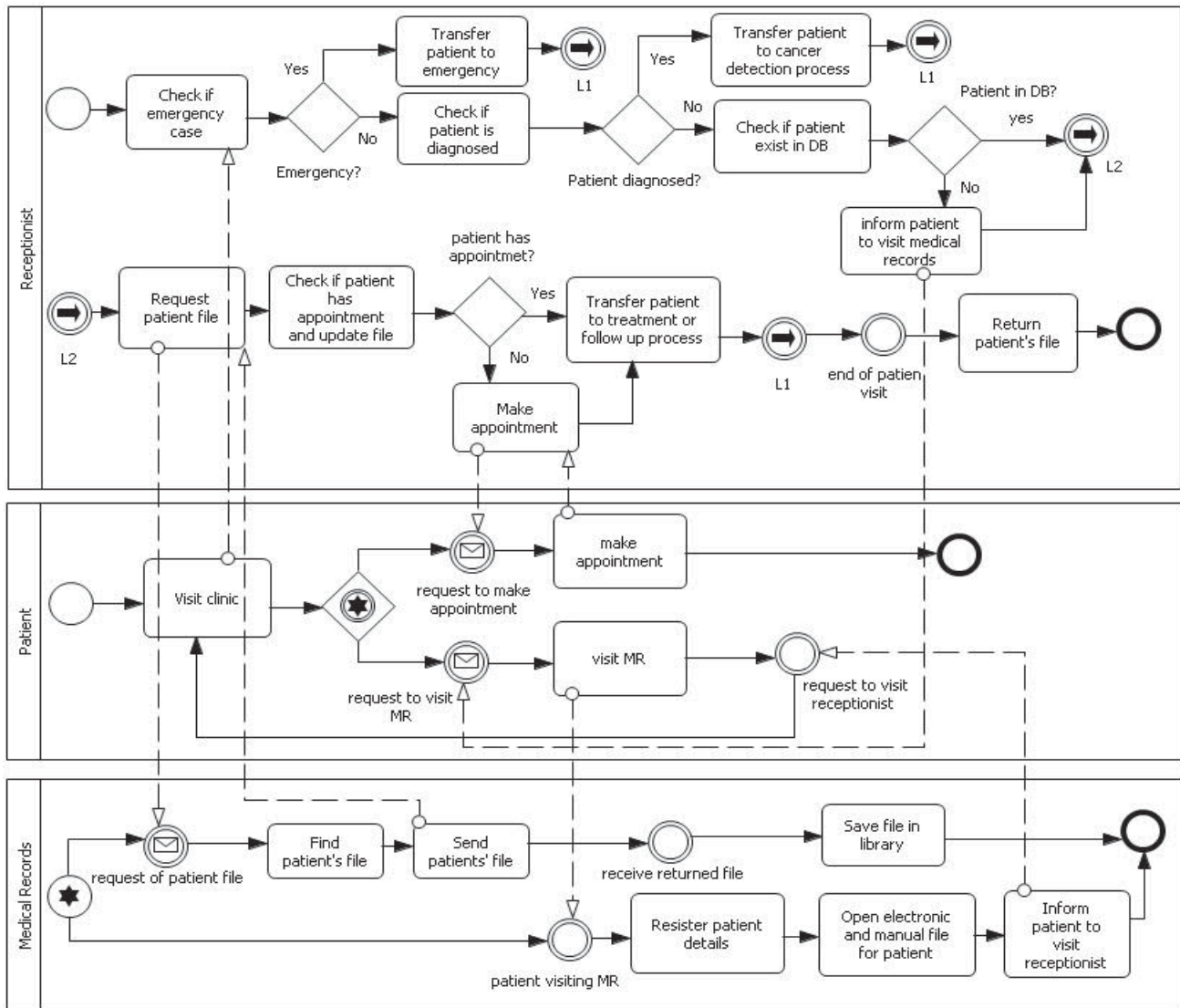


Figure 2. BPMN model of the patient reception process

References

- [1] Odeh, M., Beeson, I., Green, S., S. A. J. (2002). Modelling Processes Using RAD and UML Activity Diagrams: Exploratory Study. The 3rd International Arab Conference on Information Technology, ACIT.
- [2] AbuRub, F., Odeh, M., Beeson, I., Pheby, D., Codling, B. (2008). Modelling Healthcare Processes using Role Activity Diagramming. International Journal Of Modelling And Simulation 28 (2) 147-155.
- [3] Juric, M., Pant, K (2008). Business Process Driven SOA using BPMN and BPEL, Packt Publishing, 1st Edition.
- [4] Business process modelling notation specification, OMG, <http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf>, 2006. Accessed on March 8, 2009 at 20:15.
- [5] Polanè, G., Rozman, T. (2009). Business Process Modelling Notation (BPMN) Poster, <http://bpmn.itposter.net>. Accessed on February 20 at 18:30.
- [6] Ould, M.A. (1995). Business Processes – modelling and analysis for re-engineering and improvement, John Wiley and Sons, Chichester, 1995.
- [7] Odeh, M., Kamm, R. (2003). Bridging the Gap between Business Process Models and Systems, *Information and Software Technology*, Special Edition on Modelling Organisational Processes 45 (15) 1053-1060.

Author Biography

Rana Yousef, a PhD student in Software Engineering at the University of the West of England-Bristol. She received her MSc. in Computer Science from The University of Jordan, where she was a teacher assistant and earned a scholarship to pursue her PhD study. Her main research interest is in service oriented architectures and specifically in generating service models for SOA-based systems. To this extent, her PhD research is focused on developing a methodology for deriving a service model from a BPA.

Mohammed Odeh, Ph.D., leads the Software Engineering Research Group (SERG) in the Centre for Complex and Cooperative Systems (University of the West of England). He has more than 24 years experience in the field of software engineering with fourteen years in the banking industry. His research interests are focused on: service oriented software engineering models and processes, model-based software cost estimation, knowledge and business process driven requirements engineering with particular interest in bridging the gap between process, information, and knowledge models. He has over 50 journal, conference and book chapter publications. Dr. Odeh has been supervising a number of Ph.D. research students with successful completions. Also, he has been a co-investigator on three EU funded projects and a principal investigator on recent industrial knowledge transfer and research collaboration with Airbus. Dr. Odeh is the UWE-Bristol lead investigator on the OntoREM project with Airbus.

David Coward, Ph.D., is Head of the Department of Computer Science at the University of the West of England. He has 30 years experience in Computing and Software Engineering predominantly spent in an academic environment. He is passionate about teaching software design and programming to novices. His research interests include software testing, software metrics and cost estimation, and software design. Recently he has supervised Ph.D. students investigating evolutionary software design and cost estimation modelling.

Ahmad A. Sharieh, Ph.D., is Dean of Sur University College and used to be Dean of King Abdullah School for Information Technology at The University of Jordan. Professor Sharieh published articles in journals (24), in conferences (18), and authored and prepared books (14). He gained grant for eight research projects from UJ and Europe. He developed several software systems such as: Teaching Sign Language, e-learning Modelling and Simulation, and Online (Automated) Exams. He is on the editorial board of several journals and conferences and a referee of several others. He has been supervising a number of Master and Ph.D. research students. His research areas are in: Distributing Systems, Expert Systems, E-Government, E-Learning, Parallel Processing, Pattern Recognition, Software Engineering, Wire/Wireless Communication, Modelling and Simulation.