A Model Driven Engineering based method for scheduling analysis

Yessine Hadj kacem
CES, ENIS Soukra km 3,5
B.P.:w 3038 Sfax TUNISIA
yessine.hadjkacem@ieee.org

Adel Mahfoudhi
CES, ENIS Soukra km 3,5
B.P.:w 3038 Sfax TUNISIA
adel.mahfoudhi@fss.rnu.tn

Walid Karamti
CES, ENIS Soukra km 3,5
B.P.:w 3038 Sfax TUNISIA
walid.karamti@yahoo.fr

Mohamed Abid
CES, ENIS Soukra km 3,5
B.P.:w 3038 Sfax TUNISIA
mohamed.abid@enis.rnu.tn

***Abstract*— MDE can be used to model and analyze ERTS. However, scheduling problems are not totally covered using this approach. Traditionally, the design of these systems is limited to the characterization of the architecture and the application. Since MDE neglects task scheduling, designers are obliged to use simulation methods.**

This paper presents a scheduling analysis method for real time systems at an early stage. The proposed approach adopts the MDE concept based UML models and rule transformation in order to find a feasible schedule that satisfies timing constraints.

I. INTRODUCTION

ERTS (Embedded Real time systems) are in a continued interaction with their external environment and are constrained by time. Such systems have become increasingly complex, especially with their distributed aspects (parallel operation on various machines), where the need for using reliable techniques of scheduling emerges. Bad function or bug may cause economic and human catastrophes. Thus, the anomaly absence at simulation stage cannot confirm the non-appearance of bugs.

Therefore, to protect such systems from problems and failure, it is necessary to implement techniques intended to make reliable the development process of the real-time applications, from their design as well as checking. This allows designers to specify systems with precision, and to check the required properties of their behaviour.

Techniques founded on high level abstraction could reduce the problem impact. They could make systems reliable at a preliminary stage, and they constitute an important research field. These techniques aim to start from an abstract model annotated with a set of constraints in order to specify the properties required by the user and to check that the system specification satisfies the expressed constraints.

Currently and in the same vein, much research has been done simultaneously on MDE (Model Driven Engineering) and ERTS (Embedded Real time Systems) modeling. The use of models and model transformations make the development process more costly. Therefore, MDE represents a viable solution to increase the ERTS complexity. It helps to generate automatically embedded software from system level specification.

It is relatively easy to map ERTS problems onto MDE. However, the application of model engineering based analysis techniques to a scheduling problem represented by UML (Unified Modeling Language) models is far from trivial.

At present, designers use simulation or test feasibility or calculation of response time approaches for scheduling analysis [11]. Simulation is based on the simulation of the task progress during one system period. It checks that all the authorities of tasks respect their deadline. It can treat scheduling policies or task models which are difficult to analyze mathematically. Calculating response times applies a recurrence formula which calculates deadlines of execution on an interval that contains the greatest response times of the considered task which is met. Feasibility test which applies a formula decides the feasibility of task scheduling according to which characteristics appear to be the simplest method to implement with a low calculation complexity.

This paper proposes an approach that supports the scheduling analysis starting from high level design, and can be integrated in MDE process.

The remainder of this paper is organized as follows. Section two provides a brief discussion about related work. The proposed scheduling methodology is shown in section three where the scheduling analysis and the test feasibility are introduced through transformation rules. Section four describes progression of tasks over time. Finally, a summary and future works are given.

II. RELATED WORK

Our research results can be located in the context of two areas of related works: Scheduling Analysis methods at high abstraction levels and analysis approaches based on MDE.

Both [10] and [14] apply SPT (Scheduling, Performance and Timing) profile to apply RMA (Rate Monotonic Analysis). They use a collaboration diagram in order to demonstrate the use of UML real-time profile elements for schedulability analysis. Stereotypes allow quantifying key attributes of real time constraints. The <<SASituation>> and the <<SAschedRes>> stereotypes provide a context for schedulability analysis. Unfortunately, the used diagram is not sufficient to mention shared resource, i.e., dependent tasks. The response time and the utilization factor of the processor are not calculated. Besides, this problem is not solved with [8] in spite of the extension with analysis specific context and non functional properties.

978-1-4244-3477-0/08/$25.00 ©2008 IEEE

[1] proposes an approach for formal and simulation based performance analysis with UML2/SysML. Its method detects the communication in order to synchronize the control flow of the corresponding instances and to make the relationship explicit. It determines a global timing behaviour preset constraints. It also detects potential conflicts on the shared communication resources according to target architecture. It is true that this method evaluates system models at an early design stage but it is limited to performance analysis and does not tackle scheduling analysis aspect.

In [15], the formal method B is used in order to validate opted models. The authors start with an abstract specification from the scheduler, and refines using successive stages to take into account the time characteristics of the automats. They check the hierarchy of refinements by proving the various generated obligations. Other researches look for using model checking, especially Petri Nets. For example in [17], schedulability can be analysed using an extension of Petri NET. The authors map tasks, resources and constraints onto places and transitions of timed Petri net.

III. SCHEDULING APPROACH

A. Overview

Since transformations between models are the key elements of MDE, in this section the used models are reviewed. In fact, our research lies within the framework of the high level modeling of the RTOS. As a matter of fact, RTOS proposes various services and guarantees tasks execution and communication. These real-time services called primitives have various natures namely the management of tasks, shared resources, time, task communication and interruption. In our study, we just focus on task management which consists in controlling the task state which can have the following values: created, ready, suspended, waiting and running.

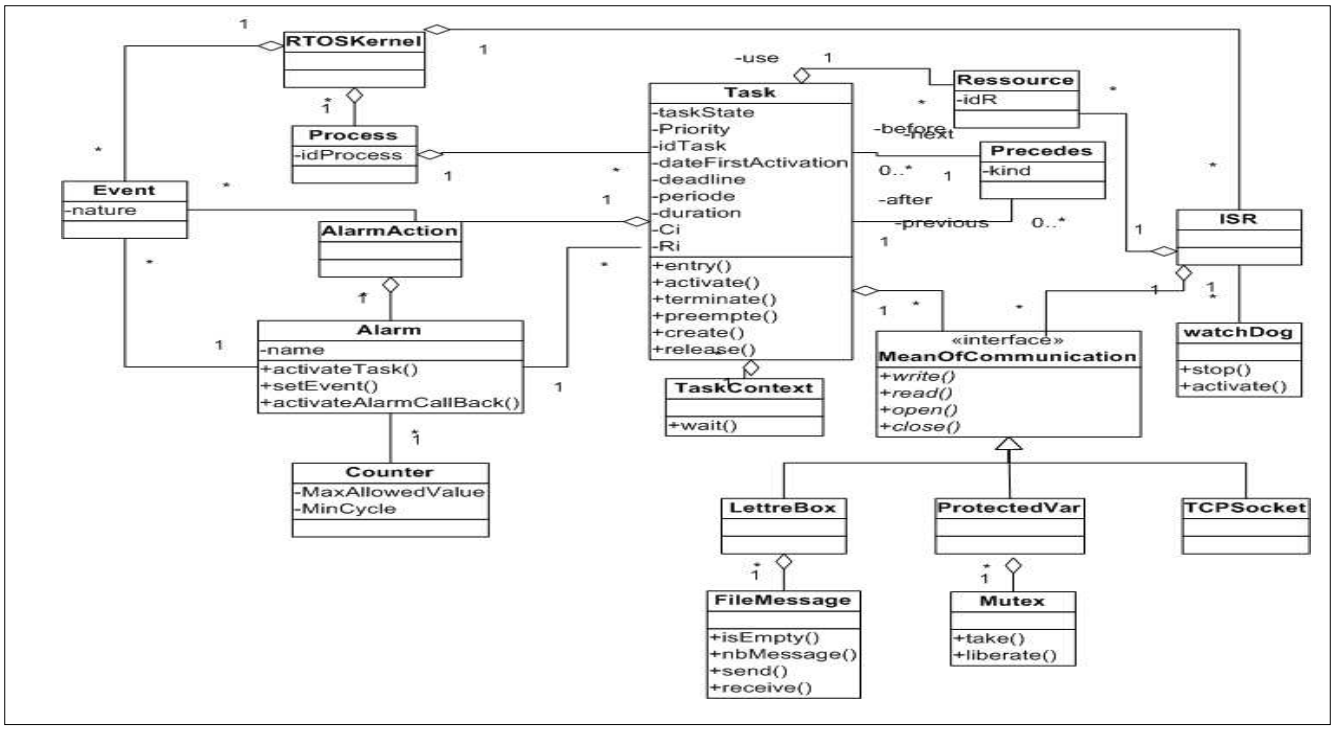

Figure 1. RTOS structure model

As a result, the current transformation takes two models specified by UML: source and scheduling models. For the source model, the RTOS structure in which the major entities are the task and the event is described. Each task may have internal attributes as shown in Figure 1. To specify the internal behaviour, the statecharts UML diagrams are used. Semantic variant points related to statecharts [2, 17] are defined using reification and enumeration techniques in order to create scheduling model. The obtained model presented by Figure 2 must cover a simple and safe scheduler required by RTOS.

To define the semantics and implement the statecharts, the approach proposed by [5], based on the enumeration and reification is adopted. This solution is based on the integration of design patterns [7] for the re-use of existing and testing software components, rather than the creation of new models for the implementation of the statecharts. It should be noted that structure model and scheduling model resulting from statecharts implementation are more detailed in [18].

B. Scheduling analysis of independent tasks

Our ultimate aim is indeed to allow the completeness of the existing Schedulability modeling sub-profiles in the pertinent aspects, while leaving them limited in the exploitation of stereotypes. For this reason, it is necessary to integrate ample scheduling analysis conditions at early stage to guarantee that the temporal requirements will be satisfied. This can be ensured during transforming models. Transformation rules can support

mathematic formulas, test the feasibility of scheduling tasks and illustrate the progression of tasks over time.

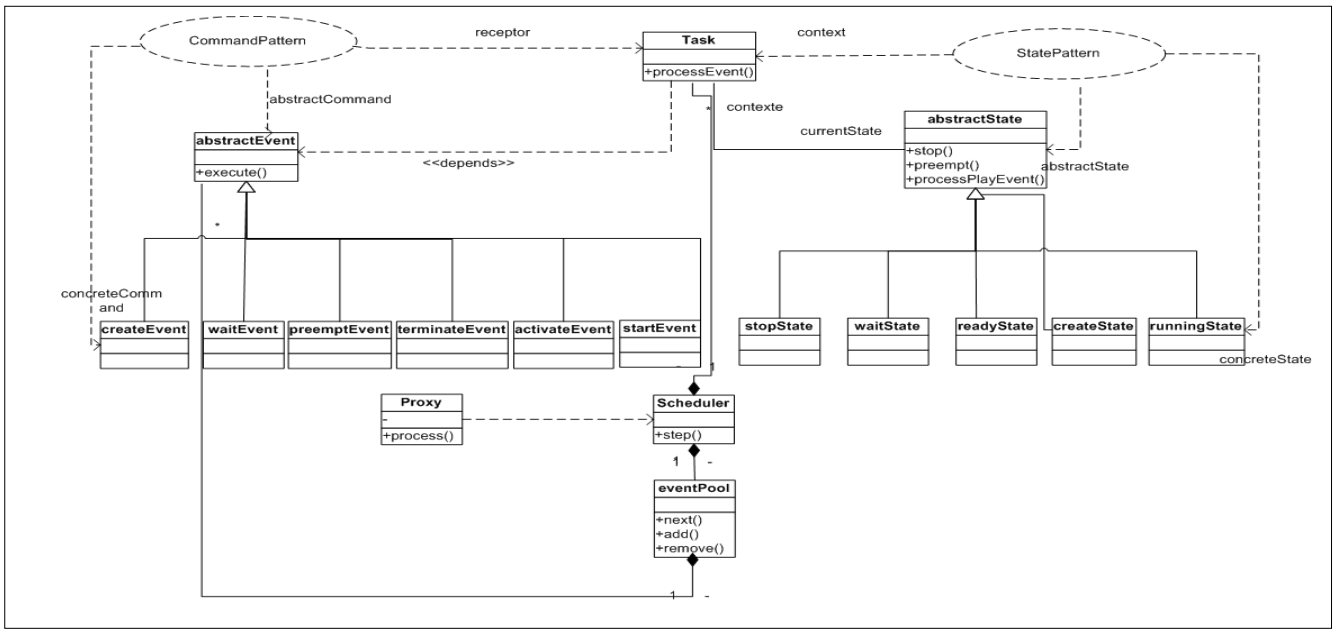

Figure 2. Scheduling model

For scheduling test strategy, we improve the method invoked by [9] using transformation rules. So, the static HPF (High priority first) scheduling policy has also been adopted.

For the specification and implementation of model transformation, the Atlas Transformation Language [6] (ATL) is used. This model transformation language uses a set of rules and helpers specified on Object Constraint Language2 (OCL). ATL specifies how target model elements are created for each matched source model element and how features are instantiated.

Like [8], tasks are scheduled statically with Highest Priority First (HPF). To check scheduling, two approaches dominate: the use of the Rate Monotonic Analysis [4] or Exact Analysis [13]. So, a first condition is primly tested and represented by the equation (1):

$$U = \sum_{i=1}^{n} \frac{C_i}{P_i} \leq 1 \qquad (1)$$

Where

- U: processor utilization factor ;
- Ci: execution time of task i;
- Pi: period of task i
- n : number of tasks

The sufficient condition is tested if the necessary one fails. It is presented by the following equation:

$$U \leq n * (2^{\frac{1}{n}} - 1) \qquad (2)$$

List 1 lists a set of these previous tests specified in ATL.

```
-- First test related to equation (1)
helper context RTOSStructure!Task1 def: U():
String=
       RTOSStructure!Task1.allInstances()
-
>collect(e|e.Ci.toInteger()/e.periode.toReal())-
>sum() ->toString();
helper context RTOSStructure!Task1 def:
Condition(x:String):Boolean=
       if self.Somme().toReal()<= 1;
       then true
       else false
       endif;
-- Second test related to equation (2)
helper context RTOSStructure!Task1 def:
ContrainteRM(x:String):String=
       x.toReal()*(2.exp(1/x.toReal()-
1)).toString();
helper context RTOSStructure!Task1 def:
ConditionRM(x:String):Boolean=
       if
self.Somme().toReal()<=self.ContrainteRM(x).toRe
al();
       then true
       else false
       endif;
-- calculating tasks numbers
helper context RTOSStructure!Task1 def:
taskNumbers():String=
       RTOSStructure!Task1.allInstances()-
>collect(e|e.idTask)
->size().toString();
```

Listing 1. Scheduling analysis with RMA

While the RMA condition is very restrictive, system schedulability can be tested according to Exact Analysis [13]

by calculating the worst case response time (WCRT). WCRT is calculated as follows

$$\forall T_i \in hp(i), \exists R_i \le D_i tq R_i = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{P_j} \right\rceil * Cj \quad [13] \ (3)$$

Where:

- HP(i) : set of tasks with higher priority comparing to task i;
- Di : deadline

In this case, only tasks with higher priority than Ti are instantiated in the target model. Ri is then calculated by the rules and the helpers shown in listing 2. Indeed, Exact Analysis considers that all tasks arrive simultaneously in the system, and if they respect their first deadline (the first response time is lower than the deadline of all tasks i), all the other deadlines will be respected.

```
-- Selecting tasks able to be instanced in the
target model
helper context RTOSStructure!Task1
def:ident_priority_max:String=
RTOSStructure!Task1.allInstances()
-> collect(a|a.priority.toInteger()).asSet()-
>select(t|t >=0)
-
>subSequence(RTOSSchudeler!Task.allInstances()-
>
       collect(f|f.priority.toInteger())
                ).last.toString();
helper context RTOSStructure!Task1
def:select_ident:Boolean=
       if self.priority.toInteger()=self.
ident_priority_max (x).tointeger()
       then true else false endif;
helper context RTOSStructure!Task1 def: Summ():
String=
       RTOSSchdeler!Task.allInstances()
->collect(e| ((
RTOSSchudeler!Task.allInstances()->
collect(a|a.Ri.toReal()).asSet()->select(t|t
>=0).last  )
               /e.periode.toReal()
               )*e.Ci.toReal())->sum()
->toString();
-- Ri calculating and Comparison between Ri and
Di
helper context RTOSStructure!Task1
def:Compar:String=
RTOSSchudeler!Task.allInstances()->
collect(a|a.Di.toReal()- a.Ri.toReal())
->select(t|t<0).size().toString();
--Final test
helper context RTOSStructure!Task1
def:Condition_EXACT: Boolean=
if self.Compar.toReal()=0
then true
else false
endif;
```

Listing 2. Scheduling analysis with Exact Analysis

After the previous assumptions that consider tasks as independent, it should be taken into account that tasks are usually dependent on one another. They collaborate together in order to make system function. They generally exchange data or share resources. So, they are faced with precedence constraints.

C. Scheduling analysis of dependent tasks

For independent tasks, scheduling analysis with MDE decreases calculating complexity on high level abstraction. Concerning dependent tasks, the contribution of MDE proves to be more important. In fact, the use of the Resource entity in the source model facilitates particularly the management of resource. The Precedes entity and the definition of operational semantics help the dependence management between tasks as well.

```
-- Identifying consuming tasks
helper context RTOSStructure!Task1
def:ident_TConsm: Boolean=
if Task1!before->including(self) then true
else false
endif;
-- loading tasks in suspended class
rule ToSuspended{
       from
            s :
RTOSStructure!Task1(s.ident_TConsm)
       to
       w : RTOSSchudeler!SuspendedState(
              idTask <- s.idTask,
                     priority <- s.priority,
                     kind <- s.before.kind
              )
}
-- calculating PiR
helper context RTOSStructure!Task1
def:priority_max:String=
RTOSStructure!Task1.allInstances()
-> collect(a|a.priority.toInteger()).asSet()
         ->select(t|t >=0).last.toString();
-- calculating Pseuil
helper context RTOSStructure!Task1
def:Pseuil:String=
       RTOSStructure!Task1.allInstances()
       ->
collect(a|a.priority.toInteger()).asSet()
       ->select(t|t >=0).last +1.toString();
-- identifying resource used by a task a t
helper context RTOSStructure!Task1
def:ident_Taskrun(t:String):String=
       RTOSSchudeler!RunninyState.allInstances(
)
       -
>select(e|datactivation.toInteger()=t.toInteger
())
       -> collect(a|a.idTask).last;
helper context RTOSStructure!Task1
def:ident_Ressrun(x:String):String=
       if self.idTask=self.ident_Taskrun(x)
       then Task1!use.idR
       else false
       endif;
```

Listing 3. Scheduling analysis of dependant tasks

In order to do that, we are initially interested in identifying the tasks which require the execution of the previous tasks so as to be executed. Listing 3 specifies the other steps able to manage shared resource. So, like [9], the running tasks and its priority are selected. Then, the resource used by this task is calculated. After that, all tasks that are waiting for the identified resource are selected. Finally, the Priority Ceiling Protocol [13] (PCP) is used to avoid unbounded priority and to support synchronization. Thanks to the association between the Task, Precedes and Resource, this step is ensured easily compared to other approaches.

IV. TASKS SCHEDULING BASED ON RULE TRANSFORMATION

The adopted approach helps us not only to check if a set of tasks can be scheduled, but also to show the automaton progression. Indeed, after verifying the previously-described tests, the source model can be translated to the scheduler model by specifying rule transformation. The generated code during this step is written in Extensible Markup Language (XMI). The resulting code can be translated to any real time programming rules by just defining the Meta Model associated to the select programming language.

The previous section proves that MDE can analyze schedulability considering different kinds of tasks. In the current section, we are limited to independent and periodic tasks with fixed priority. Therefore, after executing scheduling analysis, all tasks are instantiated in the target model. After that, the tasks that have the create state are loaded and only task with highest priority will be running. At each instant, there is a test of a task with expired period and highest priority. Task state is updated at each instance; its progress value is also incremented when running, the deadline respect always remains important.

V. CONCLUSION

We have used an RTOS model structure as source model and considered the scheduler model as the target one. Transformations between models were achieved using the ATL language. At this level, test feasibilities are carried out considering real-time constraints and inter-task relation. Analysis is an imperative concern considering critical time systems.

The proposed method that integrates simultaneously scheduling and model driven engineering could reduce the complexity of design phase, it checks the feasibility of tasks scheduling at early stage and on high level abstraction. It also gives an idea about the progression of task state over time. This can reduce software complexity as well.

Further works still remain to be done; the source model namely task entities can be checked with formal methods such as the extension of Petri Net. In order to map UML model to a Petri Net format, we need to define the Meta model conformed to the Petri Net for scheduling analysis like in [3]. The used models can also be annotated applying UML profiles particularly MARTE (Modeling and Analysis of Real-Time and Embedded systems) [16]. This way guarantees an iterative analysis process based on MDE.

REFERENCES

[1] Alexander Viehl, Timo Schönwald, Oliver Bringmann, Wolfgang Rosenstiel: Formal performance analysis and simulation of UML/SysML models for ESL design. DATE 2006: 242-247

[2] A. Cuccuru, C. Mraidha, F. Terrier, S. Gérard. Templatable Metamodels for Semantic Variation Points. ECMDA-FA 2007: 68-82

[3] Benoit Combemale, Pierre-Loïc Garoche, Xavier Crégut, Xavier Thirioux, F. Vernadat. Towards a Formal Verification of Process Model's Properties - SimplePDL and TOCL Case Study. International Conference on Enterprise Information Systems (ICEIS 2007), Funchal, Madeira - Portugal, 12/06/07-16/06/07, INSTICC Press, p. 80-89, juin 2007

[4] C. L. Liu , James W. Layland, Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment, Journal of the ACM (JACM), v.20 n.1, p.46-61, Jan. 1973

[5] F. Chauvel and J. Jézéquel. Code generation from UML models with semantic variation points. In S. Kent L. Briand, editor, Proceedings of MODELS/UML'2005, volume 3713 of LNCS, pages --, Montego Bay, Jamaica, October 2005. Springer

[6] Frédéric Jouault and Ivan Kurtev. Transforming models with ATL. In Proceedings of the Model Transformations in Practice Workshop at MoDELS2005, 2005.

[7] Gamma, Erich, Helm, Richard, Johnson, Ralph, and Vlissides, John. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Longman Publishing Co., Inc., 1995.

[8] H. Espinoza, H., J. Medina, H. Dubois, S. Gerard and F. Terrier, Towards a UML-basedvModeling Standard for Schedulability Analysis of Real-time Systems, International Workshop MARTES, MoDELS/UML 2006. Oct. 2006. Genova, Italie.

[9] H. Tmar, Jean-Philippe Diguet, Abdenour Azzedine, Mohamed Abid, Jean Luc Philippe: RTDT: A static QoS manager, RT scheduling, HW/SW partitioning CAD tool. Microelectronics Journal 37(11): 1208-1219 (2006)

[10] Hossein Saiedian, Srikrishnan Raguraman: Using UML-Based Rate Monotonic Analysis to Predict Schedulability. IEEE Computer 37(10): 56-63 (2004)

[11] J. Migge, A Jean-Marie, N. Navet, "Timing Analysis of Compound Scheduling Policies : Application to Posix1003.1b", Journal of Scheduling, Springer Verlag, vol. 6, n°5, pp457-482, 2003.

[12] M. Spuri, J.A. Stankovic "How to Integrate Precedence Constraints and Shared Resources in Real-Time Scheduling," IEEE Trans. Computers 43(12) : 1407-1412, 1994.

[13] M.Joseph, P.Pandya, Finding response time in a real-time system, Ieee Design and Test of Computers 29 (5) (1986) 390–395.

[14] María Cruz Valiente, Gonzalo Génova, Jesús Carretero: UML 2.0 Notation for Modeling Real Time Task Scheduling. Journal of Object Technology 5(4): 91-105 (2006)

[15] O. Nasr, J.-P. Bodeveix, M. Filali, andM. Rached. Verification of a Scheduler in B Through a Timed Automata Specification. In Software Engeneering Track in The 21st Annual ACM Symposium on Applied Computing (SAC'06), Dijon, pages 1800–1801. ACM, 23-27 April 2006.

[16] OMG Document Number: ptc/07-08-04. A UML Profile for MARTE, Beta 1 OMG Adopted Specification, August 2007

[17] W. Aalst. Petri net based scheduling. Computing Science Reports 95/23, Eindhoven University of Technology, Eindhoven, 1995.

[18] Y. Hadj Kacem, A. Mahfoudhi, H.Tmar, M. Abid. Towards the Automatic Generation of Real Time Operating Systems Applying UML/MDA.The Fifth International Workshop on the Applications of UML/MDA to Software Systems co-located with The International Conference on Software Engineering Research and Practice (SERP'08) July 14-17, 2008, Las Vegas, Nevada, USA