Semantic Web Technology as a Basis for Planning and Scheduling Systems

Bernd Schattenberg, Steffen Balzer, Susanne Biundo
Dept. of Artificial Intelligence
Mission Planning

- Crisis Management Support: THW missions at River Oder and Danube
- Size and complexity demand for system support for strategic decision support, for training, etc.

Application properties:
- Incomplete, mostly procedural knowledge
- Resource usage mission critical: time, personnel...
Planning and scheduling software systems in mission critical applications should feature:

- Declarative, automated system configuration and verification
  - ...for deployment and maintenance
- Scalability, including transparency with respect to system distribution, access mechanisms, concurrency, etc.
  - ...for computational power on demand, ubiquitousness
- Standards compliance
  - ...for interfacing services and environments, re-use of libraries and 3rd-party components, etc.
A Formal Framework

The *Panda* approach performs hybrid planning: HTN combined with POCL planning

- **Action representation:** primitive and complex tasks
  
  \[ t(\bar{r}) = (\text{prec}(t(\bar{r})), \text{add}(t(\bar{r})), \text{del}(t(\bar{r}))) \]

- **Plans and task networks** \((\text{TE}, \prec, \text{VC}, \text{CL})\)

- **Methods relate complex tasks and task networks:**
  - **Implementations** \(m = (t(\bar{r}), d)\)

- **Problem specification** \((d, T, M, \text{Sinit}, \text{Sgoal})\)

- **Solution criteria:** plan obtained form \(d\) by *plan refinements* and primitive, constraint sets consistent, and causal support complete and un-threatened
Planning by Refinement

- **Flaws**: explicit representation of solution criteria violations and deficiencies in the plan
  - E.g.: causal threat \((\text{Threat}, \{\langle \text{te}_i, \phi, \text{te}_j \rangle, \text{te}_k \})\)

- **Modifications**: explicit representation of plan refinement steps - elementary additions and deletions
  - E.g.: adding an ordering constraint
    \((\text{AddOrdConstr}, \{\oplus (\text{te}_i \prec \text{te}_j)\})\)

- **Modification triggering function**: which modification class addresses which flaw class, e.g.
  \(\alpha(F_{\text{Threat}}) = M_{\text{ExpandTask}} \cup M_{\text{AddOrdConstr}} \cup M_{\text{AddVarConstr}}\)
The Planning Process

- **Modification cycle**
  - Phase 1: make expertises
  - Phase 2: detect flaws
  - Phase 3: compute modifications
  - Phase 4: modify blackboard

- **Assistant cycle**
  - Strategy
  - Assistant

- **Planning Strategy**
  - Selects modifications for execution and plans
  - Independent from Inspectors and Constructors
The Architecture

- Modules should be flexibly configurable and robust
  - Agent technology: BlueJADE
- System should be transparently scalable
  - Application server middleware: JBoss
- System configuration and „verification“
  - Semantic web ontologies and reasoning facilities: DAML and RACER
- Components should be standards compliant
  - FIPA, J2EE, and W3C
Application server **JBoss** delivers

- Location transparency
- Scalability transparency
- Access transparency
- Handles concurrency
- Provides messaging service, etc.

Integrates **BlueJADE** multi-agent framework as a service
BlueJade: putting a MAS under J2EE control

Agent distribution, remote access, etc.
Lingua Franca: DAML

- DAML
  - Expressive knowledge representation language
  - Powerful reasoner support available (here: RACER)
  - Suitable content language for FIPA ACL
  - Standardized by the W3C

- Used for *agent communication* and representation of *system configuration knowledge*
A DAML ontology describes
- Agents to deploy (Inspectors, Constructors, Assistants, and Strategy)
- Agent computation products
- Flaw - Modification relationships

RACER derives
- Concrete implementations
- Inspector - Constructor assignment
- Missing implementations, unassigned Constructors...
Conclusions

- Architecture implements a proper formal framework for hybrid planning using standardized middleware and knowledge representation techniques.
- The presented approach is a platform for implementing and evaluating planning techniques and strategies that:
  - effectively supports distribution and concurrency;
  - performs a knowledge-based configuration;
  - allows planning process validation by Petri-Net transformation.