

Organizing Mobile Software Applications Using Mobile Containers

High level software abstractions are needed to isolate mobile applications from fast changing middleware

Organize mobile application functional logic as components that live inside a mobile container

The container hides the middleware technical concerns in a centralized place. Porting a family of mobile applications to new middleware would require ideally porting only the container

The main issue for an enterprise container is scalability. For a mobile container the main issue is the performance of the container itself. The number of abstraction layers should be kept at minimum

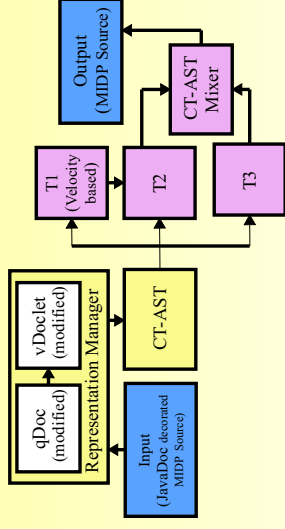
Solution: A generative based framework with transformation support at language level

MobCon Transformer Framework

A Java based framework for J2ME MIDP 2.0 applications

Concerns addressed: Data Persistence, Image Adaptation, Screen Management, Encryption of Network Messages, Session and Context. Some of the concerns generate server side code as well as client mobile code

Transformer written as Velocity scripts. Workflow defined as XML dependency file. Transformers work with a CT-AST internal representation of code. Mergers combine the output of different transformers



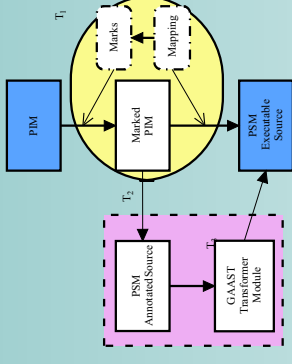
MobRay Demo with Traceability Concern



Scenario: A doctor uses his PDA to view remotely X-Ray images of patients and send back comments

```
[exec] METHOD: choosePatientCG Action [app, @log]
[exec] METHOD: retrieveEntry [app, @log]
[exec] mobcon.message.MessageHandler: Getting message from server
[exec] METHOD: setDoc [app, @log]
[exec] METHOD: callF_patient [app, @log]
[exec] METHOD: callSI_patientName [app, @log]
```

Transforming with Generalized and Annotated AST API-s Annotated AST



Tags supported at language level can be used for generation

GA-AST representation of the code AST annotated with tags. Language support for:

- annotations of arbitrary program elements,
- explicit meta-representation of programs that is accessible in a programmatic way

Explicit tags can be seen as a weak form of AOP

Hello World Example in MobCon

Tags are used to decorate elements of interest. The logical container is responsible for inserting concerns code into tagged components

```
/**
 * @scr
 */
public class HelloWorld
    extends MIDlet implements CommandListener
{
    /**
     * @scr.label "Hello"
     * @scr.firstDisplay
     * @scr.exitButton
     * @scr.textField testField
     * private Form form;
     */
    /**
     * @scr.label "First Application"
     * @scr.string "Hello World"
     * private TextField testField;
     */
}
```



Vasian Cepa
cepa@informatik-tu-darmstadt.de
Darmstadt University of Technology