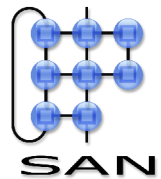


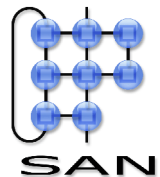
System Integrity in ...



Outline



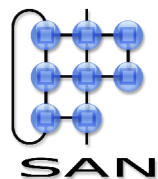
- **Introduction**
 - Problem Domain
 - Problem
 - Approach
 - Model Based Integrity Management
- **Discuss Models used for Integrity Management**
 - Models of components
 - Models of terminals
- **Discuss Integrity Management**
 - Monitoring
 - Diagnosis
 - Repairing





Problem domain

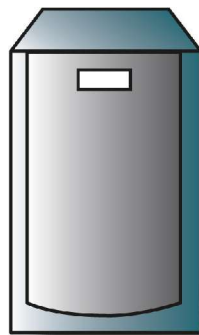
High Volume Consumer Electronics



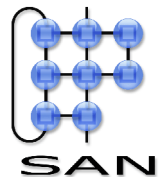
The Problem



Maintain integrity of the software on a consumer terminal in the period that the device is owned and used by the consumer.



Terminal
Manager



The Approach

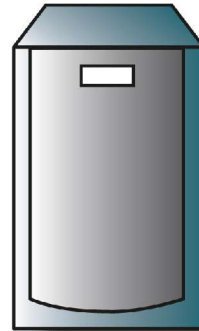


Terminal Management based on a **model** describing the **current configuration**.



Terminal

- Externalize 'Self' Model
- Facilities for Remote Management



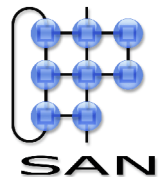
Terminal
Manager

- Monitoring
- Diagnosis
- Repairing
 - *Repair Script Generation*
 - *Repair Script Execution*

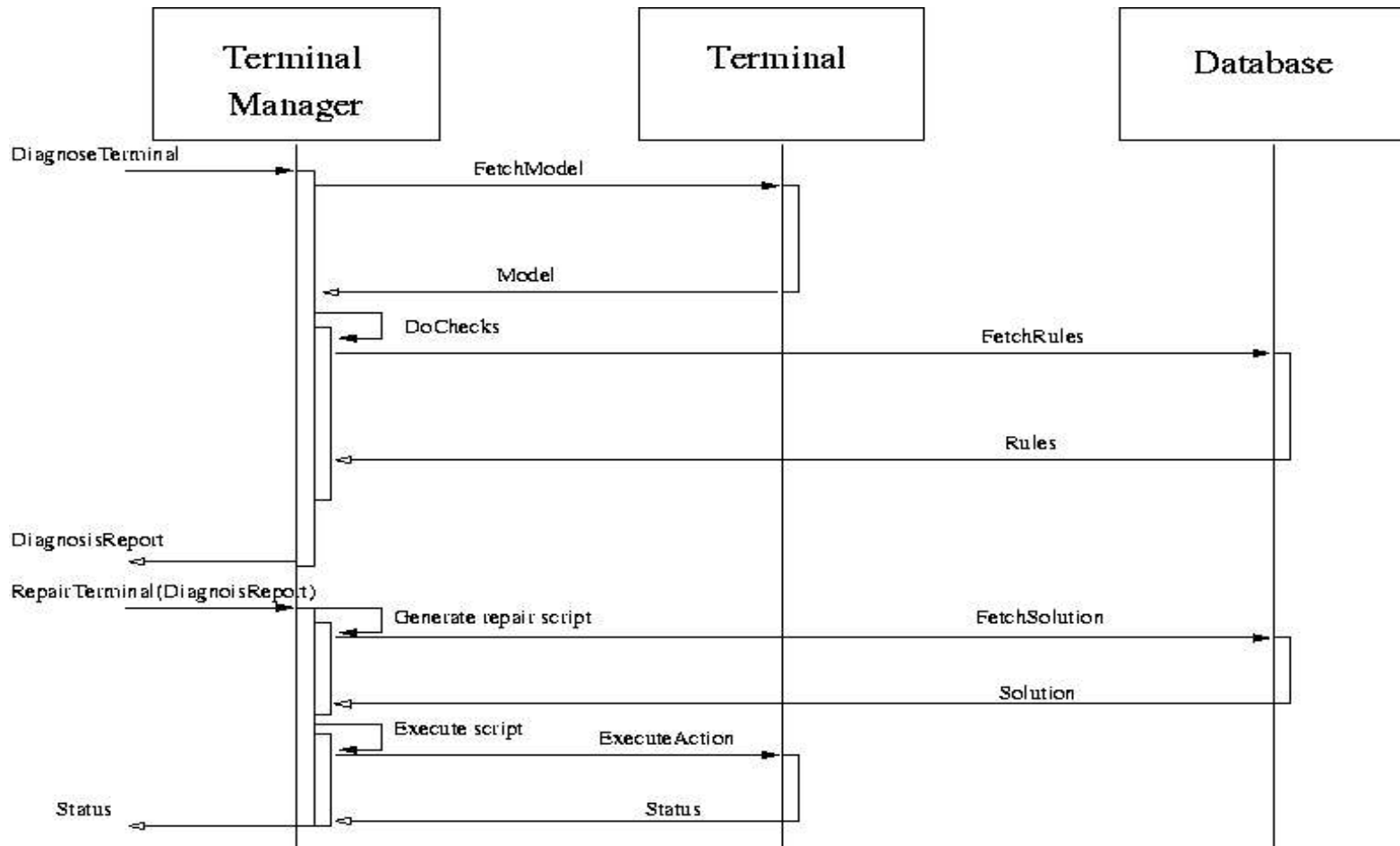


Database

- Provide Rules
- Provide Solutions

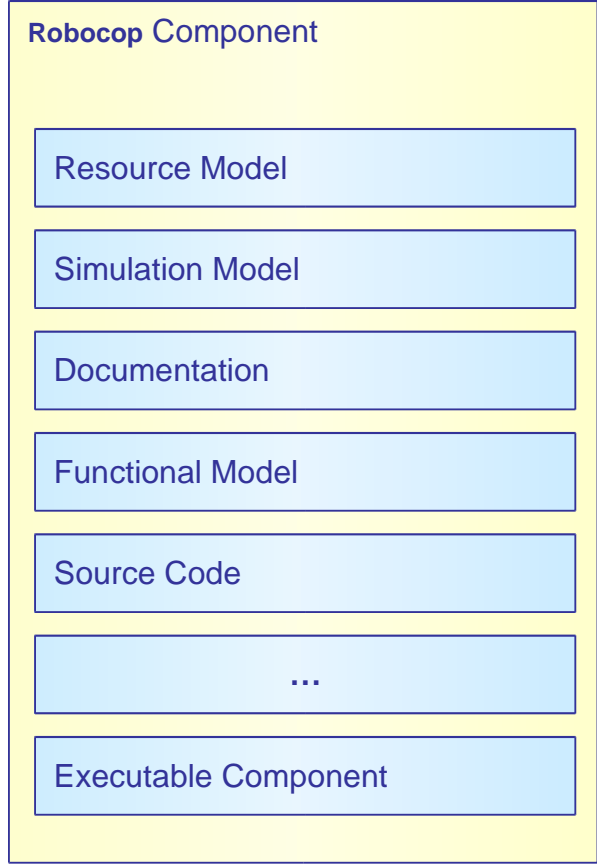


Example Scenario





Model of Robocop Component

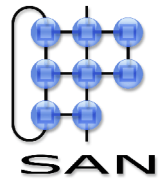


$RC = P(M) \times P(M \times M \times T)$
 $M = EM \cup BM \cup RM \cup \dots$
 $T = MODELTYPE \times MODELTYPE \times NAME$

we assume there is a function:

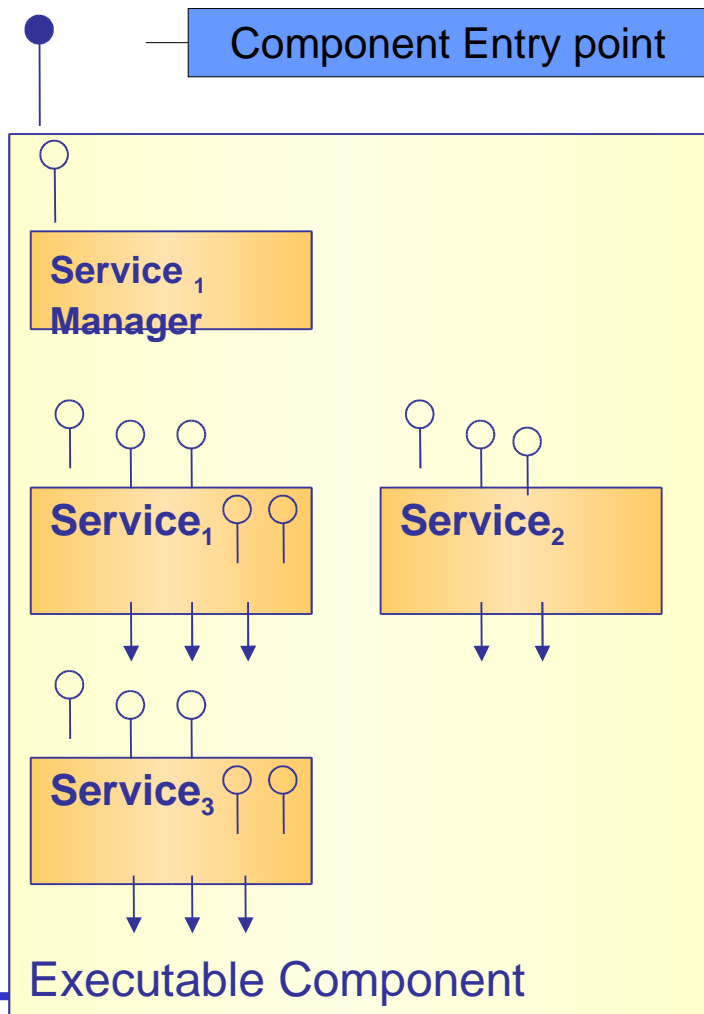
$$\phi : M \rightarrow MODELTYPE$$

$rc = \langle \{em, rm, bm\}, \{ \langle rm, em, t_1 \rangle, \langle bm, em, t_2 \rangle \} \rangle$
 $t_1 = \langle T_{RM}, T_{EM}, 'resource\ model\ of' \rangle$
 $t_2 = \langle T_{BM}, T_{EM}, 'behavior\ model\ of' \rangle$



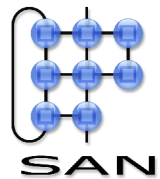


Robocop Executable Component



EM = P(S)
 S = P(P) x P(P)
 P = NAME x I
 NAME = STRING
 I = P(O)
 O = set of operations o

em = {s₁, ..., s_n}
 s₁ = <{p₁, ..., p_m}, {p_{m+1}, ..., p₁}>
 ...
 p₁ = <'player', i₁>
 ...
 i₁ = {o₁, ..., o_k}
 ...





Resource Model

RM = $R \rightarrow P(O_{imp} \rightarrow \mathbb{N})$
 O_{imp} = **S x O**
R = **set of resources r**

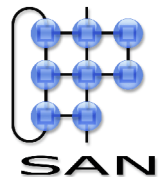
$rm(\text{memory}) = \{claim_{mem}, release_{mem}\}$

$claim_{mem}(\langle s_1, o_1 \rangle) = 100$

$release_{mem}(\langle s_1, o_1 \rangle) = 75$

$rm(\text{cpu}) = \{claim_{cpu}\}$

$claim_{cpu}(\langle s_1, o_1 \rangle) = 2500$



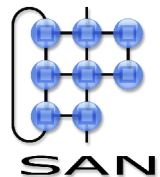
Behavior Model



$$\mathbf{BM} = \mathbf{O}_{imp} \rightarrow \mathbf{O}^*$$

$$\mathbf{O}_{imp} = \mathbf{S} \times \mathbf{O}$$

$$\text{bm}(\langle s_1, o_1 \rangle) = o_x i o_y i o_z$$



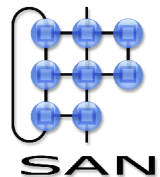
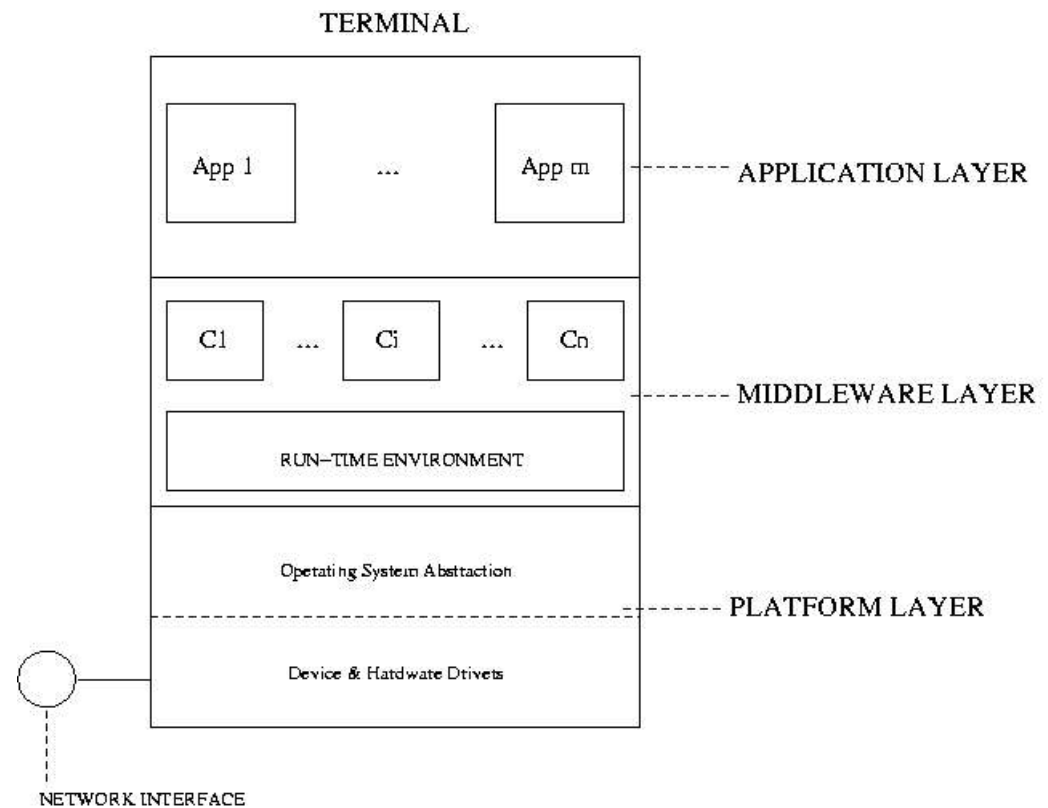
Model of a Terminal



$$\mathbf{T} = \mathbf{AL} \times \mathbf{ML} \times \mathbf{PL}$$

$$t = \langle a_1, m_1, p_1 \rangle$$

$$t \in \mathbf{T}$$





Model of Application Layer

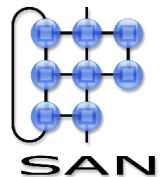
AL = **P(A)**
A = **NAME x VERSION x D x STRUCTURE**
NAME = **STRING**
VERSION = **STRING**
D = **P(S)**
STRUCTURE = **P(SI) x P(B)**
SI = **S x NAME**
B = **SI x P x SI x P**

$a_1 = \{a_1, \dots, a_n\} \in AL$

$a_1 = \langle \text{'tic tac toe'}, \text{'0.0.1'}, d_1, \text{structure}_1 \rangle \in A$

$a_2 = \langle \text{'agenda'}, \text{'0.0.2'}, d_2, \text{structure}_2 \rangle \in A$

...



Model of Middleware



ML = **RUNTIME** x **P(EM)** x **P(COMPLIES)**
RUNTIME = **VERSION**
EM = **P(S)**
COMPLIES = **S x S**

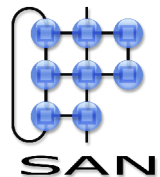
$m1 = \langle '0.2.0', \{em_1, \dots, em_n\}, \{comp_1, \dots, comp_m\} \rangle \in ML$

$em_1 = \{s_1, \dots, s_k\} \in EM$

...

$comp_1 = \langle s_x, s_y \rangle \in COMPLIES$

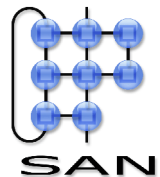
...



Model (Platform)



PL = OS x CPU x STORAGE
OS = NAME x VERSION
CPU = VENDOR x FAMILY x MODEL x SPEED x CACHE
VENDOR = STRING
FAMILY = STRING
MODEL = STRING
SPEED = IN
CACHE = IN
STORAGE = MEMORY x SWAP x P(FS)
MEMORY = IN
SWAP = IN
FS = NAME x IN



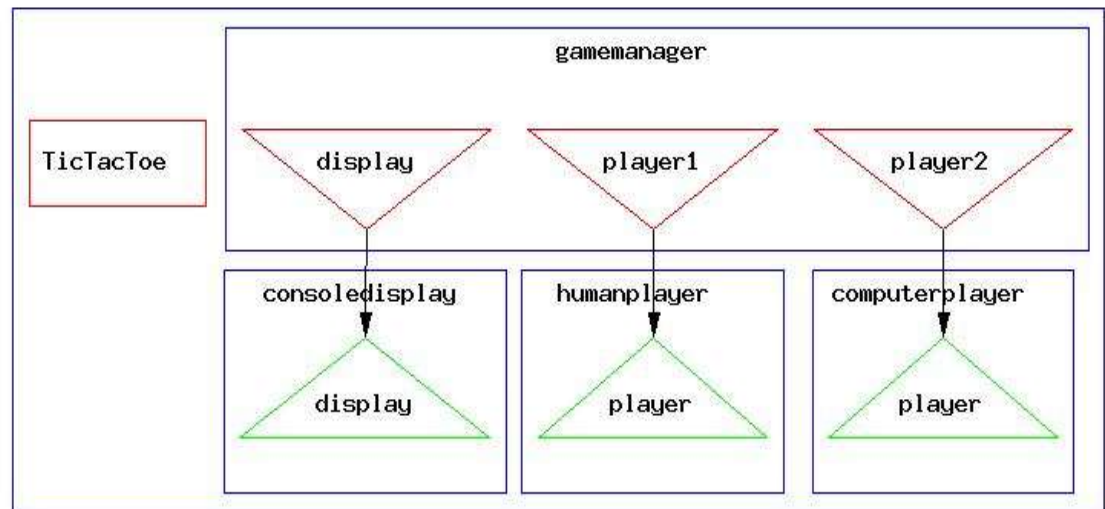
Maintaining Integrity



Monitoring:

- Show the structure of applications in the application layer
- Show the configuration of the middleware layer
- Show the type of device (platform)
- Show dependencies between applications

Example: (structure of app)



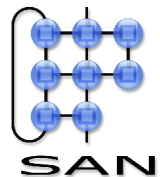
Maintaining Integrity



Diagnosis:

- Consistency checks
 - Does the middleware provide all the required services?
- Structural checks
 - Does the application adhere to design guidelines?
- Are the applications suitable for the device
 - Right platform?
 - Enough resources?
 - **We need prediction techniques !!!**

....



Maintaining Integrity



EXAMPLE CHECKS:

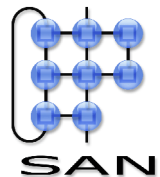
S_b = service blacklist (set of services with known faults)

$D_{BOS}(t)$ = $(\bigcup em : ml=\langle x,y,z \rangle \wedge (em \in y)) \cap S_b$
where $t=\langle x,y,z \rangle$

$P_c(t)$ = $(\bigcup em : ml=\langle x,y,z \rangle \wedge (em \in y)) \cup$
 $\{s_1 \in S \mid$
 $\langle s_1, s_2 \rangle \in (\bigcup complies : ml=\langle x,y,z \rangle \wedge (complies \in z)) \wedge$
 $s_2 \in (\bigcup em : ml=\langle x,y,z \rangle \wedge (em \in y))\}$
where $t=\langle al, ml, pl \rangle$

$S_r(t)$ = $\{s \in S \mid s \in d \wedge$
 $\langle name, version, d, structure \rangle \in al\}$
where $t=\langle al, ml, pl \rangle$

$D_{MS}(t)$ = $S_r(t) \setminus P_c(t)$



Maintaining Integrity



EXAMPLE CHECKS:

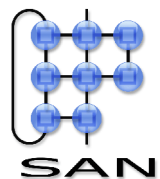
$$D_{\text{REA}} = \{a \in \text{al} \mid W(a, \text{scenario}, \text{memory}) > \text{memory}\}$$

where `scenario` is critical execution of $a \in \text{al}$
 and `storage` = $\langle \text{memory}, x, y \rangle$
 and `pl` = $\langle k, l, \text{storage} \rangle$
 and `t` = $\langle \text{al}, m_l, \text{pl} \rangle$
 for arbitrary k, l, x, y

$W(a, s, r)$ predicts the resource consumption of application a for scenario s and resource r .

For details about function W see:

J. Muskens and M. Chaudron. Prediction of run-time resource consumption in multi-task component-based software systems. Technical Report TR-117, Technische Universiteit Eindhoven, 2003

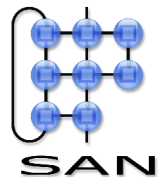


Maintaining Integrity



Repairing:

- Depends on the diagnosis
 - Blacklist of services?
 - Missing services?
 - Resource Exhaustion by Application?
 - ...



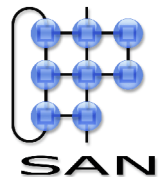
Future Work



- Check Design Guidelines
 - *Right Structure?*
 - *Right Behavior?*
 - *Right Architecture Style?*
 - Interaction model (communication & control)

- Language for easy specification of integrity constraints

- ...



My Question for You...



- How to express integrity constraints in a easy way?
 - What type of language to use ?
 - Is it possible ?

- How to express design guidelines
 - Structure ?
 - Behavior ?

