Project: DRA FRONT END FILTER PROJECT

Title: Phase 3 Theory Listings

Ref: DS/FMU/FEF/045 *Issue: Revision* : *2.1* *Date:* 5 June 2016

Status: Approved *Type:* Specification

Keywords:

Author:

Name	*Location*	*Signature*	*Date*
R. B. Jones	WIN01		

Authorisation for Issue:

Name	*Function*	*Signature*	*Date*
R.B. Jones	HAT Manager		

Abstract: A collection of listings of theories developed under Phase 3 of the FEF contract, together with an index. (for DRA Front End Filter project RSRE 1C/6130.)

Distribution: HAT FEF File
Simon Wiseman

0 DOCUMENT CONTROL

0.1 Contents List

0.2 Document Cross References

[1] DS/FMU/FEF/040. *Multi-level Formal Security Policy.* R.D. Arthan, ICL Secure Systems, WIN01.

[2] DS/FMU/FEF/042. *Multi-level Architectural Model.* R.D. Arthan, ICL Secure Systems, WIN01.

[3] DS/FMU/FEF/043. *The Labelling Property for SWORD.* R.B. Jones, ICL Secure Systems, WIN01.

[4] DS/FMU/IED/wrk057. *Examples of HOL Type Definitions.* R.D. Arthan, Lemma 1 Ltd., `http://www.lemma-one.com`.

0.3 Changes History

Issue 1.1 (11 March 1994) First draft.

Issue *Revision* : *2.1* **(5 June 2016)** Final approved version.

Issue 2.2 Removed dependency on ICL logo font

0.4 Changes Forecast

None.

1 GENERAL

1.1 Scope

This document contains listings of the theories developed under Phase 3 of the FEF contract [1, 2, 3], together with an index. The theory *wrk057*[4] is also included since its listing is not available elsewhere.

2 THE THEORY wrk057

2.1 Parents

lib_thms

2.2 Children

fef042

2.3 Constants

FinitaryRecType
$$('D \to 'T) \leftrightarrow (('D \to 'T\ \mathbb{P}) \times ('T \to \mathbb{N}) \times (('T \to 'y) \to 'D \to 'Y))$$
LocalFunctional
$$(('T \to 'y) \to 'D \to 'Y) \leftrightarrow ('D \to 'T\ \mathbb{P})$$
Tree $(\mathbb{N} \times 'a)\ LIST\ \mathbb{P}$
Unparse $(\mathbb{N} \times 'a)\ LIST \to \mathbb{N} \to (\mathbb{N} \times 'a)\ LIST$
MkTree $'a \times 'a\ TREE\ LIST \to 'a\ TREE$

2.4 Types

$'1$ **TREE**

2.5 Definitions

FinitaryRecType
$$\begin{array}{l}\vdash FinitaryRecType \\ = \{(k,\ c,\ w,\ M) \\ |OneOne\ k \\ \quad \wedge (\forall\ t \bullet \exists\ x \bullet\ c\ x \subseteq \{z | w\ z < w\ t\} \wedge t = k\ x) \\ \quad \wedge (\forall\ i\ g_1\ g_2 \\ \quad \bullet (\forall\ y \bullet\ w\ y < i \Rightarrow g_1\ y = g_2\ y) \\ \quad\quad \Rightarrow (\forall\ x \\ \quad\quad \bullet\ c\ x \subseteq \{y | w\ y < i\} \Rightarrow M\ g_1\ x = M\ g_2\ x))\}\end{array}$$
LocalFunctional
$$\begin{array}{l}\vdash LocalFunctional \\ = \{(M,\ c) \\ |\forall\ I\ g_1\ g_2 \\ \quad \bullet (\forall\ y \bullet\ y \in I \Rightarrow g_1\ y = g_2\ y) \\ \quad\quad \Rightarrow (\forall\ x \bullet\ c\ x \subseteq I \Rightarrow M\ g_1\ x = M\ g_2\ x)\}\end{array}$$
Tree
$$\begin{array}{l}\vdash Tree \\ = \bigcap \\ \quad \{A \\ \quad\quad |\forall\ lab\ trees \\ \quad\quad\quad \bullet\ Elems\ trees \subseteq A\end{array}$$

$$\Rightarrow Cons\ (\#\ trees,\ lab)\ (Flat\ trees) \in A\}$$

Unparse
$$\vdash \forall\ i\ nv\ more \bullet Unparse\ []\ i = [] \wedge Unparse\ (Cons\ nv\ more)\ i = (if\ i = 0\ then\ []\ else\ Cons\ nv\ (Unparse\ more\ ((Fst\ nv + i) - 1)))$$

TREE

tree_def
$$\vdash \exists\ f \bullet TypeDefn\ (\lambda\ t \bullet t \in Tree)\ f$$

MkTree
$$\vdash ConstSpec\ (\lambda\ MkTree' \bullet OneOne\ MkTree' \wedge (\exists\ w \bullet \forall\ t \bullet \exists\ x \bullet (\forall\ z \bullet z \in Elems\ (Snd\ x) \Rightarrow w\ z < w\ t) \wedge t = MkTree'\ x))\ MkTree$$

2.6 Theorems

⌢_empty_thm
$$\vdash \forall\ l1\ l2 \bullet l1 \frown l2 = [] \Leftrightarrow l1 = [] \wedge l2 = []$$

flat_empty_thm
$$\vdash \forall\ ls \bullet Flat\ ls = [] \Leftrightarrow Elems\ ls \subseteq \{[]\}$$

append_assoc_thm
$$\vdash \forall\ l1\ l2\ l3 \bullet (l1 \frown l2) \frown l3 = l1 \frown l2 \frown l3$$

flat_append_thm
$$\vdash \forall\ ls1\ ls2 \bullet Flat\ (ls1 \frown ls2) = Flat\ ls1 \frown Flat\ ls2$$

length_append_thm
$$\vdash \forall\ ls1\ ls2 \bullet \#\ (ls1 \frown ls2) = \#\ ls1 + \#\ ls2$$

elems_append_thm
$$\vdash \forall\ l1\ l2 \bullet Elems\ (l1 \frown l2) = Elems\ l1 \cup Elems\ l2$$

append_empty_thm
$$\vdash \forall\ l \bullet l \frown [] = l$$

append_cancel_thm
$$\vdash \forall\ l1\ l2\ l3 \bullet l1 \frown l2 = l1 \frown l3 \Leftrightarrow l2 = l3$$

map_map_id_thm
$$\vdash \forall\ f\ g\ l \bullet (\forall\ x \bullet x \in Elems\ l \Rightarrow f\ (g\ x) = x) \Rightarrow Map\ f\ (Map\ g\ l) = l$$

length_length_flat_thm
$$\vdash \forall\ ll\ l \bullet l \in Elems\ ll \Rightarrow \#\ l \leq \#\ (Flat\ ll)$$

elems_map_thm
$$\vdash \forall\ f\ l \bullet Elems\ (Map\ f\ l) = \{y | \exists\ x \bullet x \in Elems\ l \wedge f\ x = y\}$$

length_0_thm
$$\vdash \forall\ l \bullet \#\ l = 0 \Leftrightarrow l = []$$

one_one_left_inv_thm

$$\vdash \forall f\bullet \mathit{OneOne}\ f \Rightarrow (\exists\ g\bullet \forall\ x\bullet\ g\ (f\ x) = x)$$

onto_right_inv_thm

$$\vdash \forall f\bullet \mathit{Onto}\ f \Rightarrow (\exists\ g\bullet \forall\ y\bullet\ f\ (g\ y) = y)$$

one_one_onto_inv_thm

$$\begin{array}{l}\vdash \forall\ f\\ \quad\bullet\ \mathit{OneOne}\ f \wedge \mathit{Onto}\ f\\ \qquad\Rightarrow (\exists\ g\bullet\ (\forall\ x\bullet\ g\ (f\ x) = x) \wedge (\forall\ y\bullet\ f\ (g\ y) = y))\end{array}$$

fin_rec_type_induction_thm

$$\begin{array}{l}\vdash \forall\ k\ c\ w\ M\\ \quad\bullet\ (k,\ c,\ w,\ M) \in \mathit{FinitaryRecType}\\ \qquad\Rightarrow (\forall\ X\bullet\ (\forall\ x\bullet\ c\ x \subseteq X \Rightarrow k\ x \in X) \Rightarrow (\forall\ t\bullet\ t \in X))\end{array}$$

fin_rec_type_induction_thm1

$$\begin{array}{l}\vdash \forall\ k\ c\ w\ M\\ \quad\bullet\ (k,\ c,\ w,\ M) \in \mathit{FinitaryRecType}\\ \qquad\Rightarrow (\forall\ P\\ \qquad\bullet\ (\forall\ x\bullet\ (\forall\ t\bullet\ t \in c\ x \Rightarrow P\ t) \Rightarrow P\ (k\ x))\\ \qquad\quad\Rightarrow (\forall\ t\bullet\ P\ t))\end{array}$$

fin_rec_type_prim_rec_lemma1

$$\begin{array}{l}\vdash \forall\ k\ c\ w\\ \quad\bullet\ (k,\ c,\ w,\ M) \in \mathit{FinitaryRecType}\\ \qquad\Rightarrow (\exists\ \delta\\ \qquad\bullet\ (\forall\ x\bullet\ \delta\ (k\ x) = x)\\ \qquad\quad\wedge\ (\forall\ y\bullet\ k\ (\delta\ y) = y)\\ \qquad\quad\wedge\ (\forall\ d\\ \qquad\quad\bullet\ \exists\ H\\ \qquad\qquad\bullet\ (\forall\ y\bullet\ H\ 0\ y = d\ (\delta\ y,\ M\ \mathit{Arbitrary}\ (\delta\ y)))\\ \qquad\qquad\quad\wedge\ (\forall\ i\ y\\ \qquad\qquad\quad\bullet\ H\ (i + 1)\ y\\ \qquad\qquad\qquad= (\mathit{if}\ w\ y \leq i\\ \qquad\qquad\qquad\quad\mathit{then}\ H\ i\ y\\ \qquad\qquad\qquad\quad\mathit{else}\ d\ (\delta\ y,\ M\ (H\ i)\ (\delta\ y))))\\ \qquad\qquad\quad\wedge\ (\forall\ j\ y\bullet\ H\ (w\ y + j)\ y = H\ (w\ y)\ y)))\end{array}$$

fin_rec_type_prim_rec_exists_thm

$$\begin{array}{l}\vdash \forall\ k\ c\ w\ M\\ \quad\bullet\ (k,\ c,\ w,\ M) \in \mathit{FinitaryRecType}\\ \qquad\Rightarrow (\forall\ d\bullet\ \exists\ h\bullet\ \forall\ x\bullet\ h\ (k\ x) = d\ (x,\ M\ h\ x))\end{array}$$

fin_rec_type_prim_rec_unique_thm

$$\begin{array}{l}\vdash \forall\ k\ c\ w\ M\\ \quad\bullet\ (k,\ c,\ w,\ M) \in \mathit{FinitaryRecType}\\ \qquad\Rightarrow (\forall\ d\ h_1\ h_2\\ \qquad\bullet\ (\forall\ x\bullet\ h_1\ (k\ x) = d\ (x,\ M\ h_1\ x))\\ \qquad\quad\wedge\ (\forall\ x\bullet\ h_2\ (k\ x) = d\ (x,\ M\ h_2\ x))\\ \qquad\quad\Rightarrow h_1 = h_2)\end{array}$$

fin_rec_type_prim_rec_thm

$$\begin{array}{l}\vdash \forall\ k\ c\ w\ M\\ \quad\bullet\ (k,\ c,\ w,\ M) \in \mathit{FinitaryRecType}\\ \qquad\Rightarrow (\forall\ d\bullet\ \exists_1\ h\bullet\ \forall\ x\bullet\ h\ (k\ x) = d\ x\ (M\ h\ x))\end{array}$$

local_functional_thm

$$\vdash \forall\ c\ w\ M \bullet (M,\ c) \in \mathit{LocalFunctional} \Rightarrow (\forall\ i\ g_1\ g_2 \bullet (\forall\ y \bullet w\ y < i \Rightarrow g_1\ y = g_2\ y) \Rightarrow (\forall\ x \bullet c\ x \subseteq \{y | w\ y < i\} \Rightarrow M\ g_1\ x = M\ g_2\ x))$$

i_local_thm $\vdash ((\lambda\ f \bullet f),\ (\lambda\ x \bullet \{x\})) \in \mathit{LocalFunctional}$

k_local_thm $\vdash ((\lambda\ f\ x \bullet \{\}),\ (\lambda\ x \bullet \{x\})) \in \mathit{LocalFunctional}$

×_local_thm

$$\vdash \forall\ M_1\ c_1\ M_2\ c_2 \bullet (M_1,\ c_1) \in \mathit{LocalFunctional} \wedge (M_2,\ c_2) \in \mathit{LocalFunctional} \Rightarrow ((\lambda\ f\ (x,\ y) \bullet (M_1\ f\ x,\ M_2\ f\ y)),\ (\lambda\ (x,\ y) \bullet c_1\ x \cup c_2\ y)) \in \mathit{LocalFunctional}$$

sum_local_thm

$$\vdash \forall\ M_1\ c_1\ M_2\ c_2 \bullet (M_1,\ c_1) \in \mathit{LocalFunctional} \wedge (M_2,\ c_2) \in \mathit{LocalFunctional} \Rightarrow ((\lambda\ f\ x \bullet \mathit{if}\ \mathit{IsL}\ x\ \mathit{then}\ M_1\ f\ (\mathit{OutL}\ x)\ \mathit{else}\ M_2\ f\ (\mathit{OutR}\ x)),\ (\lambda\ x \bullet \mathit{if}\ \mathit{IsL}\ x\ \mathit{then}\ c_1\ (\mathit{OutL}\ x)\ \mathit{else}\ c_2\ (\mathit{OutR}\ x))) \in \mathit{LocalFunctional}$$

list_local_thm

$$\vdash \forall\ M\ c \bullet (M,\ c) \in \mathit{LocalFunctional} \Rightarrow ((\lambda\ f \bullet \mathit{Map}\ (M\ f)),\ (\lambda\ x \bullet \bigcup (\mathit{Elems}\ (\mathit{Map}\ c\ x)))) \in \mathit{LocalFunctional}$$

list_local_thm1

$$\vdash (\mathit{Map},\ \mathit{Elems}) \in \mathit{LocalFunctional}$$

o_snd_local_thm

$$\vdash \forall\ M\ c \bullet (M,\ c) \in \mathit{LocalFunctional} \Rightarrow ((\lambda\ f \bullet M\ f\ o\ \mathit{Snd}),\ c\ o\ \mathit{Snd}) \in \mathit{LocalFunctional}$$

o_fst_local_thm

$$\vdash \forall\ M\ c \bullet (M,\ c) \in \mathit{LocalFunctional} \Rightarrow ((\lambda\ f \bullet M\ f\ o\ \mathit{Fst}),\ c\ o\ \mathit{Fst}) \in \mathit{LocalFunctional}$$

tree_induction_lemma1

$$\vdash \forall\ X \bullet (\forall\ x\ ts \bullet \mathit{Elems}\ ts \subseteq X \Rightarrow \mathit{Cons}\ (\#\ ts,\ x)\ (\mathit{Flat}\ ts) \in X) \Rightarrow \mathit{Tree} \subseteq X$$

tree_induction_lemma2

$\vdash \forall\ x\ ts$
$\bullet\ Elems\ ts \subseteq Tree \Rightarrow Cons\ (\#\ ts,\ x)\ (Flat\ ts) \in Tree$

tree_induction_lemma

$\vdash \forall\ X$
$\bullet\ (\forall\ x\ ts$
$\bullet\ Elems\ ts \subseteq Tree \cap X$
$\Rightarrow Cons\ (\#\ ts,\ x)\ (Flat\ ts) \in X)$
$\Rightarrow Tree \subseteq X$

tree_induction_tac_lemma

$\vdash \forall\ P$
$\bullet\ (\forall\ x\ ts$
$\bullet\ (\forall\ t\bullet\ t \in Elems\ ts \Rightarrow t \in Tree \wedge P\ t)$
$\Rightarrow P\ (Cons\ (\#\ ts,\ x)\ (Flat\ ts)))$
$\Rightarrow (\forall\ t\bullet\ t \in Tree \Rightarrow P\ t)$

tree_cases_lemma

$\vdash \forall\ t$
$\bullet\ t \in Tree$
$\Rightarrow (\exists\ x\ ts$
$\bullet\ Elems\ ts \subseteq Tree \wedge t = Cons\ (\#\ ts,\ x)\ (Flat\ ts))$

¬_empty_list_tree_lemma

$\vdash \neg\ [] \in Tree$

unparse_thm $\vdash \forall\ ts\ more$
$\bullet\ Elems\ ts \subseteq Tree$
$\Rightarrow Unparse\ (Flat\ ts \frown more)\ (\#\ ts) = Flat\ ts$

unparse_thm1 $\vdash \forall\ t\ more\bullet\ t \in Tree \Rightarrow Unparse\ (t \frown more)\ 1 = t$

tree_cases_lemma1

$\vdash \forall\ t$
$\bullet\ t \in Tree$
$\Rightarrow (\exists_1\ (x,\ ts)$
$\bullet\ Elems\ ts \subseteq Tree \wedge t = Cons\ (\#\ ts,\ x)\ (Flat\ ts))$

leaf_is_a_tree_thm

$\vdash \forall\ x\bullet\ [(0,\ x)] \in Tree$

MkTree_consistent

$\vdash Consistent$
$(\lambda\ MkTree'$
$\bullet\ OneOne\ MkTree'$
$\wedge (\exists\ w$
$\bullet \forall\ t$
$\bullet \exists\ x$
$\bullet\ (\forall\ z\bullet\ z \in Elems\ (Snd\ x) \Rightarrow w\ z < w\ t)$
$\wedge\ t = MkTree'\ x))$

tree_local_thm

$\vdash ((\lambda\ f\bullet\ Map\ f\ o\ Snd),\ Elems\ o\ Snd) \in LocalFunctional$

tree_fin_rec_thm

$\vdash \exists\ w$
$\bullet\ (MkTree,\ Elems\ o\ Snd,\ w,\ (\lambda\ f\bullet\ Map\ f\ o\ Snd))$
$\in FinitaryRecType$

tree_induction_thm

$$\vdash \forall\ P \bullet (\forall\ x \bullet (\forall\ t \bullet\ t \in \mathit{Elems}\ (\mathit{Snd}\ x) \Rightarrow P\ t) \Rightarrow P\ (\mathit{MkTree}\ x)) \Rightarrow (\forall\ t \bullet\ P\ t)$$

tree_prim_rec_thm

$$\vdash \forall\ d \bullet \exists_1\ h \bullet \forall\ x \bullet\ h\ (\mathit{MkTree}\ x) = d\ x\ (\mathit{Map}\ h\ (\mathit{Snd}\ x))$$

3 THE THEORY fef040

3.1 Parents

fef010 fef003

3.2 Children

fef042

3.3 Constants

$\$\downarrow$	$Class \to Class\ \mathbb{P}$
$\$\uparrow$	$Class \to Class\ \mathbb{P}$
$\$^{\perp}$	$Class\ \mathbb{P} \to Class\ \mathbb{P}$
Equivalence	$('a \leftrightarrow 'a)\ \mathbb{P}$
IndexedEquiv	$(Class\ \mathbb{P} \to 'a \leftrightarrow 'a)\ \mathbb{P}$
LiftRel	$(Class \to 'a \leftrightarrow 'a) \to Class\ \mathbb{P} \to 'a \leftrightarrow 'a$
Independent	$(Class\ \mathbb{P} \to 'a \leftrightarrow 'a)\ \mathbb{P}$
x_ml_secure	$(Class\ \mathbb{P} \to 'I \leftrightarrow 'I) \to (Class\ \mathbb{P} \to 'O \leftrightarrow 'O) \to ('I \to 'O)\ \mathbb{P}$
ml_secure	$(Class \to 'I \leftrightarrow 'I) \to (Class \to 'O \leftrightarrow 'O) \to ('I \to 'O)\ \mathbb{P}$
Influenced	$(Class\ \mathbb{P} \to 'a \leftrightarrow 'a) \to 'a \to ('a \to 'b) \to Class\ \mathbb{P}$
ObservedValue	$('a, 'b)\ OBSERVATION \to 'a \to Class \times 'b$
SameLabVal	$Class \to (Class \times 'b) \leftrightarrow (Class \times 'b)$
BoundedObs	$(Class\ \mathbb{P} \to 'a \leftrightarrow 'a) \to Class \leftrightarrow (('a \to Class) \times ('a \to 'b))$

3.4 Type Abbreviations

$('a, 'b)$ **OBSERVATION**

$('a, 'b)\ OBSERVATION$

3.5 Fixity

Postfix 300: $\downarrow$ $\uparrow$ $^{\perp}$

3.6 Definitions

$\downarrow$ $\vdash \forall\ c\bullet\ c\downarrow = \{d | c\ \mathit{dominates}\ d\}$

$\uparrow$ $\vdash \forall\ c\bullet\ c\uparrow = \{d | d\ \mathit{dominates}\ c\}$

$\perp$ $\vdash \forall\ A$
$\bullet\ A^{\perp}$
$= \{c$
$| \forall\ d\bullet\ d \in A \Rightarrow \neg\ c\ \mathit{dominates}\ d \wedge \neg\ d\ \mathit{dominates}\ c\}$

Equivalence $\vdash \mathit{Equivalence} = \mathit{Reflexive} \cap \mathit{Symmetric} \cap \mathit{Transitive}$

IndexedEquiv $\vdash \mathit{IndexedEquiv}$
$= \{s$
$| (\forall\ A\bullet\ s\ A \in \mathit{Equivalence})$
$\wedge\ (\forall\ A\ B\bullet\ A \subseteq B \Rightarrow s\ B \subseteq s\ A)\}$

LiftRel $\vdash \forall\ R\bullet\ \mathit{LiftRel}\ R = (\lambda\ A\bullet \bigcap \{r | \exists\ c\bullet\ c \in A \wedge r = R\ c\})$

Independent $\vdash \mathit{Independent}$
$= \{s$
$| \forall\ A\ x$
$\bullet\ (\exists\ y\bullet\ \neg\ (x, y) \in s\ A)$
$\Rightarrow (\exists\ z\bullet\ \neg\ (x, z) \in s\ A \wedge (x, z) \in s\ (A^{\perp}))\}$

x_ml_secure $\vdash \forall\ s_I\ s_O\ b$
$\bullet\ b \in x\_ml\_secure\ s_I\ s_O$
$\Leftrightarrow s_I \in \mathit{IndexedEquiv}$
$\wedge\ s_O \in \mathit{IndexedEquiv}$
$\wedge\ (\forall\ c\ i_1\ i_2$
$\bullet\ (i_1, i_2) \in s_I\ (c\downarrow)$
$\Rightarrow (b\ i_1, b\ i_2) \in s_O\ (c\downarrow))$

ml_secure $\vdash \forall\ r_I\ r_O$
$\bullet\ \mathit{ml\_secure}\ r_I\ r_O$
$= x\_ml\_secure\ (\mathit{LiftRel}\ r_I)\ (\mathit{LiftRel}\ r_O)$

Influenced $\vdash \forall\ s\ x\ f$
$\bullet\ \mathit{Influenced}\ s\ x\ f$
$= \{c$
$| \exists\ y\bullet\ (x, y) \in s\ (\sim (c\uparrow)) \wedge \neg\ f\ x = f\ y\}$

ObservedValue
$\vdash \forall\ c\ C\ V\ x\bullet\ \mathit{ObservedValue}\ (c, C, V)\ x = (C\ x, V\ x)$

SameLabVal $\vdash \forall\ c$
$\bullet\ \mathit{SameLabVal}\ c$
$= \{((c_1, v_1), c_2, v_2)$
$| c_1 = c_2 \wedge (c\ \mathit{dominates}\ c_1 \Rightarrow v_1 = v_2)\}$

BoundedObs $\vdash \forall\ s$
$\bullet\ \mathit{BoundedObs}\ s$
$= \{(c, C, V)$
$| \forall\ x$
$\bullet\ \mathit{Influenced}\ s\ x\ C \subseteq c\downarrow$
$\wedge\ \mathit{Influenced}\ s\ x\ V \subseteq C\ x\downarrow\}$

3.7 Theorems

⌈_⌉_thm $\vdash \forall\ l\ a\ b \bullet\ l \upharpoonright a \upharpoonright b = l \upharpoonright (a \cap b)$

not_lattice_top_thm
$\vdash \forall\ c$
$\bullet \neg\ c = lattice\_top \Rightarrow \neg\ lattice\_top = lattice\_bottom$

up_down_thm1 $\vdash lattice\_bottom\uparrow = Universe$
$\wedge\ lattice\_top\downarrow = Universe$

up_down_thm2 $\vdash \forall\ c$
$\bullet\ c \in c\downarrow$
$\wedge\ c \in c\uparrow$
$\wedge\ lattice\_bottom \in c\downarrow$
$\wedge\ lattice\_top \in c\uparrow$

up_down_clauses
$\vdash \forall\ c$
$\bullet\ (lattice\_bottom\uparrow = Universe$
$\wedge\ lattice\_top\downarrow = Universe)$
$\wedge\ c \in c\downarrow$
$\wedge\ c \in c\uparrow$
$\wedge\ lattice\_bottom \in c\downarrow$
$\wedge\ lattice\_top \in c\uparrow$

up_down_thm3 $\vdash \forall\ c\ d$
$\bullet \neg\ d\ dominates\ c \Rightarrow d\downarrow \subseteq \sim (c\uparrow)$

lift_rel_indexed_equiv_thm
$\vdash \forall\ R$
$\bullet\ (\forall\ c \bullet\ R\ c \in Equivalence) \Rightarrow LiftRel\ R \in IndexedEquiv$

same_ins_equiv_thm
$\vdash \forall\ c \bullet\ same\_ins\ c \in Equivalence$

same_outs_equiv_thm
$\vdash \forall\ c \bullet\ same\_outs\ c \in Equivalence$

same_ins_same_outs_indexed_equiv_thm
$\vdash LiftRel\ same\_ins \in IndexedEquiv$
$\wedge\ LiftRel\ same\_outs \in IndexedEquiv$

equiv_anti_mono_lift_rel_thm
$\vdash \forall\ R$
$\bullet\ (\forall\ c \bullet\ R\ c \in Equivalence)$
$\wedge\ (\forall\ c\ d \bullet\ c\ dominates\ d \Rightarrow R\ c \subseteq R\ d)$
$\Rightarrow (\forall\ c \bullet\ LiftRel\ R\ (c\downarrow) = R\ c)$

same_ins_anti_mono_thm
$\vdash \forall\ c\ d \bullet\ c\ dominates\ d \Rightarrow same\_ins\ c \subseteq same\_ins\ d$

same_outs_anti_mono_thm
$\vdash \forall\ c\ d \bullet\ c\ dominates\ d \Rightarrow same\_outs\ c \subseteq same\_outs\ d$

lift_rel_same_ins_same_outs_down_set_thm
$\vdash \forall\ c$
$\bullet\ LiftRel\ same\_ins\ (c\downarrow) = same\_ins\ c$
$\wedge\ LiftRel\ same\_outs\ (c\downarrow) = same\_outs\ c$

thm_040_1 $\vdash \forall\ bm \bullet\ bm \in secure \Leftrightarrow bm \in ml\_secure\ same\_ins\ same\_outs$

same_lab_val_equiv_thm

$\vdash \forall\ c \bullet\ SameLabVal\ c \in Equivalence$

lift_rel_same_lab_val_equiv_thm

$\vdash\ LiftRel\ SameLabVal \in IndexedEquiv$

same_lab_val_anti_mono_thm

$\vdash \forall\ c\ d \bullet\ c\ dominates\ d \Rightarrow SameLabVal\ c \subseteq SameLabVal\ d$

lift_rel_same_lab_val_down_set_thm

$\vdash \forall\ c$
$\bullet\ LiftRel\ SameLabVal\ (c\downarrow) = SameLabVal\ c$

thm_040_2

$\vdash \forall\ s$
$\bullet\ s \in IndexedEquiv$
$\Rightarrow (\forall\ c\ C\ V$
$\bullet\ ObservedValue\ (c,\ C,\ V)$
$\in x\_ml\_secure\ s\ (LiftRel\ SameLabVal)$
$\Rightarrow (c,\ C,\ V) \in BoundedObs\ s)$

down_⋂_comp_up_thm

$\vdash \forall\ c$
$\bullet\ c\downarrow$
$= \bigcap$
$\{A | \exists\ d \bullet \neg\ c\ dominates\ d \wedge A = \sim (d\uparrow)\}$

4 THE THEORY fef042

4.1 Parents

fef036 wrk057 fef040

4.2 Children

fef043

4.3 Constants

objectRefers $Obj \rightarrow Obj\ LIST$

objectContains $Obj \rightarrow Text$

objectClass $Obj \rightarrow Class$

MkObj $Class \times Text \times Obj\ LIST \rightarrow Obj$

reqSsql $Req \rightarrow Obj$

reqClearance $Req \rightarrow Class$

MkReq $Class \rightarrow Obj \rightarrow Req$

identicalObj $Class \rightarrow Obj \leftrightarrow Obj$

identicalObjs $Class \rightarrow Obj\ LIST \leftrightarrow Obj\ LIST$

VisibleReq $Class \rightarrow Req\ \mathbb{P}$

sameRequest $Class \rightarrow Req \leftrightarrow Req$

sameRequests $Class \rightarrow Req\ LIST \leftrightarrow Req\ LIST$

VisibleOutput $Class \rightarrow Obj\ \mathbb{P}$

sameOutputs $Class \rightarrow Obj\ LIST \leftrightarrow Obj\ LIST$

SWORD_ml_secure $ML\_BEHAVIOUR\ \mathbb{P}$

Init $'State\ Machine \rightarrow 'State$

Output $'State\ Machine \rightarrow 'State \times Req \rightarrow Obj$

Next $'State\ Machine \rightarrow 'State \times Req \rightarrow 'State$

MkMachine $('State \times Req \rightarrow 'State) \rightarrow ('State \times Req \rightarrow Obj) \rightarrow 'State \rightarrow 'State\ Machine$

lift_machine $'State\ Machine \rightarrow 'State \times Req\ LIST \rightarrow Obj\ LIST \times 'State$

Behaviours $'State\ Machine \rightarrow ML\_BEHAVIOUR$

FilterObj $Class \rightarrow Obj \rightarrow Obj$

SWORD_construction $'State\ Machine \rightarrow ML\_BEHAVIOUR$

sameFilterInputs $Class \rightarrow Obj\ LIST \leftrightarrow Obj\ LIST$

FlowSecureMachine $'State\ Machine\ \mathbb{P}$

4.4 Types

Req
$'1$ **Machine**

4.5 Type Abbreviations

Text *Text*
Obj *Obj*
ML_BEHAVIOUR *ML_BEHAVIOUR*

4.6 Definitions

MkObj
objectClass
objectContains
objectRefers $\vdash$ *ConstSpec*
$(\lambda$
$(MkObj', objectClass', objectContains',$
$objectRefers')$
$\bullet \forall\ c\ t\ os$
$\bullet\ MkObj'\ (c, t, os) = MkTree\ ((c, t), os)$
$\wedge\ objectClass'\ (MkObj'\ (c, t, os)) = c$
$\wedge\ objectContains'\ (MkObj'\ (c, t, os)) = t$
$\wedge\ objectRefers'\ (MkObj'\ (c, t, os)) = os)$
$(MkObj, objectClass, objectContains, objectRefers)$
Req $\vdash \exists\ f\bullet\ TypeDefn\ (\lambda\ x\bullet\ true)\ f$
MkReq
reqClearance
reqSsql $\vdash \forall\ t\ x1\ x2$
$\bullet\ reqClearance\ (MkReq\ x1\ x2) = x1$
$\wedge\ reqSsql\ (MkReq\ x1\ x2) = x2$
$\wedge\ MkReq\ (reqClearance\ t)\ (reqSsql\ t) = t$
identicalObj $\vdash$ *ConstSpec*
$(\lambda\ identicalObj'$
$\bullet \forall\ c$
$\bullet\ identicalObj'\ c$
$= \{(o_1, o_2)$
$|\exists\ c_1\ t_1\ os_1\ c_2\ t_2\ os_2$
$\bullet\ o_1 = MkObj\ (c_1, t_1, os_1)$
$\wedge\ o_2 = MkObj\ (c_2, t_2, os_2)$
$\wedge\ c_1 = c_2$
$\wedge\ (c\ dominates\ c_1$
$\Rightarrow t_1 = t_2$
$\wedge\ \#\ os_1 = \#\ os_2$
$\wedge\ Elems\ (Combine\ os_1\ os_2)$
$\subseteq identicalObj'\ c)\})$
$identicalObj$

identicalObjs

$$\vdash \forall\ c \bullet identicalObjs\ c = \{(s_1, s_2) \mid \#\ s_1 = \#\ s_2 \wedge Elems\ (Combine\ s_1\ s_2) \subseteq identicalObj\ c\}$$

VisibleReq

$$\vdash \forall\ c \bullet VisibleReq\ c = \{r \mid c\ dominates\ objectClass\ (reqSsql\ r)\}$$

sameRequest

$$\vdash \forall\ c \bullet sameRequest\ c = \{(r_1, r_2) \mid (reqSsql\ r_1, reqSsql\ r_2) \in identicalObj\ c\}$$

sameRequests

$$\vdash \forall\ c \bullet sameRequests\ c = \{(rs_1, rs_2) \mid let\ rs_3 = rs_1 \upharpoonright VisibleReq\ c\ in\ let\ rs_4 = rs_2 \upharpoonright VisibleReq\ c\ in\ \#\ rs_3 = \#\ rs_4 \wedge Elems\ (Combine\ rs_3\ rs_4) \subseteq sameRequest\ c\}$$

VisibleOutput

$$\vdash \forall\ c \bullet VisibleOutput\ c = \{ob \mid c\ dominates\ objectClass\ ob\}$$

sameOutputs

$$\vdash \forall\ c \bullet sameOutputs\ c = \{(os_1, os_2) \mid os_1 \upharpoonright VisibleOutput\ c = os_2 \upharpoonright VisibleOutput\ c\}$$

SWORD_ml_secure

$$\vdash SWORD\_ml\_secure = ml\_secure\ sameRequests\ sameOutputs$$

Machine

$$\vdash \exists\ f \bullet TypeDefn\ (\lambda\ x \bullet true)\ f$$

MkMachine

Next

Output

Init

$$\vdash \forall\ t\ x1\ x2\ x3 \bullet Next\ (MkMachine\ x1\ x2\ x3) = x1 \wedge Output\ (MkMachine\ x1\ x2\ x3) = x2 \wedge Init\ (MkMachine\ x1\ x2\ x3) = x3 \wedge MkMachine\ (Next\ t)\ (Output\ t)\ (Init\ t) = t$$

lift_machine

$$\vdash ConstSpec\ (\lambda\ lift\_machine' \bullet \forall\ mch\ s\ r\ rl \bullet lift\_machine'\ mch\ (s, []) = ([], s) \wedge lift\_machine'\ mch\ (s, Cons\ r\ rl) = (let\ out = Output\ mch\ (s, r)\ and\ s' = Next\ mch\ (s, r)\ in\ let\ (outl, final\_state) = lift\_machine'\ mch\ (s', rl)$$

$$
\begin{array}{l}
\qquad\qquad\qquad\qquad in\ (Cons\ out\ outl,\ final\_state)))\\
\quad lift\_machine\\
\textbf{Behaviours}\quad \vdash \forall\ mch\ rs\\
\quad \bullet\ Behaviours\ mch\ rs\\
\qquad = Fst\ (lift\_machine\ mch\ (Init\ mch,\ rs))\\
\textbf{FilterObj}\quad \vdash\ ConstSpec\\
\quad (\lambda\ FilterObj'\\
\quad \bullet \forall\ c\ d\ t\ os\\
\quad \bullet\ FilterObj'\ c\ (MkObj\ (d,\ t,\ os))\\
\qquad = (if\ c\ dominates\ d\\
\qquad\quad then\ MkObj\ (d,\ t,\ Map\ (FilterObj'\ c)\ os)\\
\qquad\quad else\ MkObj\ (d,\ Arbitrary,\ Arbitrary)))\\
\quad FilterObj\\
\textbf{SWORD\_construction}\\
\quad \vdash \forall\ mch\\
\quad \bullet\ SWORD\_construction\ mch\\
\qquad = (let\ sec\_output\ (st,\ r)\\
\qquad\quad = FilterObj\\
\qquad\quad (reqClearance\ r)\\
\qquad\quad (Output\ mch\ (st,\ r))\\
\qquad in\ let\ sec\_mch\\
\qquad\quad = MkMachine\\
\qquad\quad (Next\ mch)\\
\qquad\quad sec\_output\\
\qquad\quad (Init\ mch)\ in\ Behaviours\ sec\_mch)\\
\textbf{sameFilterInputs}\\
\quad \vdash \forall\ c\\
\quad \bullet\ sameFilterInputs\ c\\
\qquad = \{(os_1,\ os_2)\\
\qquad |let\ os_3 = os_1 \upharpoonright VisibleOutput\ c\\
\qquad\quad in\ let\ os_4 = os_2 \upharpoonright VisibleOutput\ c\\
\qquad\quad in\ \#\ os_3 = \#\ os_4\\
\qquad\quad \wedge\ Elems\ (Combine\ os_3\ os_4)\\
\qquad\quad \subseteq identicalObj\ c\}\\
\textbf{FlowSecureMachine}\\
\quad \vdash\ FlowSecureMachine\\
\qquad = \{mch\\
\qquad |Behaviours\ mch\\
\qquad\quad \in ml\_secure\ sameRequests\ sameFilterInputs\}
\end{array}
$$

4.7 Theorems

mk_tree_one_one_thm

$$\vdash \forall\ x\ y\bullet\ MkTree\ x = MkTree\ y \Rightarrow x = y$$

mk_tree_onto_thm

$$\vdash \forall\ t\bullet \exists\ x\bullet\ t = MkTree\ x$$

MkObj_consistent

objectClass_consistent

objectContains_consistent
objectRefers_consistent

$$
\begin{array}{l}
\vdash \mathit{Consistent} \\
\quad (\lambda \\
\qquad (\mathit{MkObj}', \mathit{objectClass}', \mathit{objectContains}', \\
\qquad\quad \mathit{objectRefers}') \\
\qquad \bullet\ \forall\ c\ t\ os \\
\qquad\quad \bullet\ \mathit{MkObj}'\ (c, t, os) = \mathit{MkTree}\ ((c, t), os) \\
\qquad\qquad \wedge\ \mathit{objectClass}'\ (\mathit{MkObj}'\ (c, t, os)) = c \\
\qquad\qquad \wedge\ \mathit{objectContains}'\ (\mathit{MkObj}'\ (c, t, os)) = t \\
\qquad\qquad \wedge\ \mathit{objectRefers}'\ (\mathit{MkObj}'\ (c, t, os)) = os)
\end{array}
$$

obj_prim_rec_thm

$$
\begin{array}{l}
\vdash \forall\ d \\
\quad \bullet\ \exists_1\ h \\
\qquad \bullet\ \forall\ c\ t\ os \bullet\ h\ (\mathit{MkObj}\ (c, t, os)) = d\ c\ t\ (\mathit{Map}\ h\ os)
\end{array}
$$

object_clauses

$$
\begin{array}{l}
\vdash (\forall\ x \\
\quad \bullet\ \mathit{MkObj}\ x \\
\qquad = \mathit{MkTree}\ ((\mathit{Fst}\ x, \mathit{Fst}\ (\mathit{Snd}\ x)), \mathit{Snd}\ (\mathit{Snd}\ x))) \\
\quad \wedge\ (\forall\ \mathit{ctos} \\
\quad \bullet\ \mathit{objectClass}\ (\mathit{MkTree}\ \mathit{ctos}) = \mathit{Fst}\ (\mathit{Fst}\ \mathit{ctos}) \\
\qquad \wedge\ \mathit{objectContains}\ (\mathit{MkTree}\ \mathit{ctos}) = \mathit{Snd}\ (\mathit{Fst}\ \mathit{ctos}) \\
\qquad \wedge\ \mathit{objectRefers}\ (\mathit{MkTree}\ \mathit{ctos}) = \mathit{Snd}\ \mathit{ctos})
\end{array}
$$

lift_machine_consistent

$$
\begin{array}{l}
\vdash \mathit{Consistent} \\
\quad (\lambda\ \mathit{lift\_machine}' \\
\quad \bullet\ \forall\ \mathit{mch}\ s\ r\ \mathit{rl} \\
\quad \bullet\ \mathit{lift\_machine}'\ \mathit{mch}\ (s, []) = ([], s) \\
\qquad \wedge\ \mathit{lift\_machine}'\ \mathit{mch}\ (s, \mathit{Cons}\ r\ \mathit{rl}) \\
\qquad = (\mathit{let}\ \mathit{out} = \mathit{Output}\ \mathit{mch}\ (s, r) \\
\qquad\quad \mathit{and}\ s' = \mathit{Next}\ \mathit{mch}\ (s, r) \\
\qquad\quad \mathit{in}\ \mathit{let}\ (\mathit{outl}, \mathit{final\_state}) \\
\qquad\qquad = \mathit{lift\_machine}'\ \mathit{mch}\ (s', \mathit{rl}) \\
\qquad\quad \mathit{in}\ (\mathit{Cons}\ \mathit{out}\ \mathit{outl}, \mathit{final\_state})))
\end{array}
$$

identicalObj_consistent

$$
\begin{array}{l}
\vdash \mathit{Consistent} \\
\quad (\lambda\ \mathit{identicalObj}' \\
\quad \bullet\ \forall\ c \\
\quad \bullet\ \mathit{identicalObj}'\ c \\
\qquad = \{(o_1, o_2) \\
\qquad |\exists\ c_1\ t_1\ os_1\ c_2\ t_2\ os_2 \\
\qquad\quad \bullet\ o_1 = \mathit{MkObj}\ (c_1, t_1, os_1) \\
\qquad\qquad \wedge\ o_2 = \mathit{MkObj}\ (c_2, t_2, os_2) \\
\qquad\qquad \wedge\ c_1 = c_2 \\
\qquad\qquad \wedge\ (c\ \mathit{dominates}\ c_1 \\
\qquad\qquad\quad \Rightarrow t_1 = t_2 \\
\qquad\qquad\qquad \wedge\ \#\ os_1 = \#\ os_2 \\
\qquad\qquad\qquad \wedge\ \mathit{Elems}\ (\mathit{Combine}\ os_1\ os_2)
\end{array}
$$

$\subseteq$ *identicalObj'* *c*)})

FilterObj_consistent

$\vdash$ *Consistent*
(λ *FilterObj'*
• ∀ *c d t os*
• *FilterObj'* *c* (*MkObj* (*d*, *t*, *os*))
= (*if* *c dominates d*
then MkObj (*d*, *t*, *Map* (*FilterObj'* *c*) *os*)
else MkObj (*d*, *Arbitrary*, *Arbitrary*)))

elems_combine_thm

$\vdash$ ∀ *l x y*
• (*x*, *y*) ∈ *Elems* (*Combine l l*) ⇔ *y* ∈ *Elems l* ∧ *x* = *y*

elems_combine_elems_thm

$\vdash$ ∀ *l1 l2 x y*
• # *l1* = # *l2* ∧ (*x*, *y*) ∈ *Elems* (*Combine l1 l2*)
⇒ *x* ∈ *Elems l1*

elems_combine_swap_thm

$\vdash$ ∀ *l1 l2 x1 x2*
• # *l1* = # *l2* ∧ (*x1*, *x2*) ∈ *Elems* (*Combine l1 l2*)
⇒ (*x2*, *x1*) ∈ *Elems* (*Combine l2 l1*)

elems_combine_map_thm

$\vdash$ ∀ *f l1 l2 x1 x2*
• # *l1* = # *l2*
∧ (*x1*, *x2*) ∈ *Elems* (*Combine l1 l2*)
∧ *Map f l1* = *Map f l2*
⇒ *f x1* = *f x2*

elems_combine_map_thm1

$\vdash$ ∀ *f l1 l2*
• # *l1* = # *l2*
∧ (∀ *y1 y2*
• (*y1*, *y2*) ∈ *Elems* (*Combine l1 l2*)
⇒ *f y1* = *f y2*)
⇒ *Map f l1* = *Map f l2*

obj_induction_thm

$\vdash$ ∀ *P*
• (∀ *c t ts*
• (∀ *t*• *t* ∈ *Elems ts* ⇒ *P t*)
⇒ *P* (*MkObj* (*c*, *t*, *ts*)))
⇒ (∀ *t*• *P t*)

identical_obj_filter_obj_thm1

$\vdash$ ∀ *c ob1 ob2*
• (*ob1*, *ob2*) ∈ *identicalObj c*
⇒ *FilterObj c ob1* = *FilterObj c ob2*

identical_obj_filter_obj_thm2

$\vdash$ ∀ *c ob1 ob2*
• *FilterObj c ob1* = *FilterObj c ob2*
⇒ (*ob1*, *ob2*) ∈ *identicalObj c*

identical_obj_filter_obj_thm

$$\vdash \forall\ c \bullet identicalObj\ c = \{(ob1,\ ob2) | FilterObj\ c\ ob1 = FilterObj\ c\ ob2\}$$

identical_obj_refl_thm

$$\vdash \forall\ c\ ob \bullet (ob,\ ob) \in identicalObj\ c$$

identical_obj_sym_thm

$$\vdash \forall\ c\ ob1\ ob2 \bullet (ob1,\ ob2) \in identicalObj\ c \Rightarrow (ob2,\ ob1) \in identicalObj\ c$$

identical_obj_trans_thm

$$\vdash \forall\ c\ ob1\ ob2\ ob3 \bullet (ob1,\ ob2) \in identicalObj\ c \wedge (ob2,\ ob3) \in identicalObj\ c \Rightarrow (ob1,\ ob3) \in identicalObj\ c$$

identical_obj_rft_thm

$$\vdash \forall\ c \bullet identicalObj\ c \in Reflexive \wedge identicalObj\ c \in Symmetric \wedge identicalObj\ c \in Transitive$$

identical_obj_equiv_thm

$$\vdash \forall\ c \bullet identicalObj\ c \in Equivalence$$

same_requests_filter_obj_thm

$$\vdash \forall\ c \bullet sameRequests\ c = \{(rs1,\ rs2) | Map\ (FilterObj\ c\ o\ reqSsql)\ (rs1 \upharpoonright VisibleReq\ c) = Map\ (FilterObj\ c\ o\ reqSsql)\ (rs2 \upharpoonright VisibleReq\ c)\}$$

same_requests_equiv_thm

$$\vdash \forall\ c \bullet sameRequests\ c \in Equivalence$$

same_outputs_equiv_thm

$$\vdash \forall\ c \bullet sameOutputs\ c \in Equivalence$$

same_requests_indexed_equiv_thm

$$\vdash LiftRel\ sameRequests \in IndexedEquiv$$

same_outputs_indexed_equiv_thm

$$\vdash LiftRel\ sameOutputs \in IndexedEquiv$$

5 THE THEORY fef043

5.1 Parents

fef042

5.2 Constants

$\$_{\mathbf{f}}$ $'State\ Factor \to 'State\ Factor \to 'State\ Factor$

same_to_level
$Worth \to Obj \leftrightarrow Obj$

factor_level $Worth \to 'State\ Factor\ \mathbb{P}$

factor3 $'State\ Factorisation \to 'State\ Factor$

factor2 $'State\ Factorisation \to 'State\ Factor$

factor1 $'State\ Factorisation \to 'State\ Factor$

factor0 $'State\ Factorisation \to 'State\ Factor$

MkFactorisation
$'State\ Factor \to 'State\ Factor \to 'State\ Factor \to 'State\ Factor \to 'State\ Factorisation$

composite $'State\ Factorisation \to 'State\ Factor$

factor_out $'State\ Factorisation \to 'State \times Req \to Obj$

levelled_factorisation
$'State\ Factorisation\ \mathbb{P}$

factors $'State\ FactoredMachine \to 'State\ Factorisation$

machine $'State\ FactoredMachine \to 'State\ Machine$

MkFactoredMachine
$'State\ Machine \to 'State\ Factorisation \to 'State\ FactoredMachine$

well_factored
$'State\ FactoredMachine\ \mathbb{P}$

special_machine
$'State\ Machine \times 'State\ Factor \times 'State\ Factor \to Req\ LIST \to Req \to Req\ LIST \to Obj$

same_at_c_below_level
$Worth \to Class \to Obj \leftrightarrow Obj$

label_secure_to
$Worth \to 'State\ Machine \leftrightarrow ('State\ Factor \times 'State\ Factor)$

label_secure $'State\ FactoredMachine\ \mathbb{P}$

simplest_witness
$'State\ FactoredMachine$

identity_witness
$'State\ FactoredMachine$
purge_above_level
$Worth \to Class \to Obj \to Obj$

5.3 Types

$'1$ **Factorisation**
$'1$ **FactoredMachine**

5.4 Type Abbreviations

$'State$ **Factor**
$'State\ Factor$

5.5 Fixity

Right Infix 250:
$\fatsemi_{\mathbf{f}}$

5.6 Definitions

$\fatsemi_{\mathbf{f}}$ $\vdash \forall\ f1\ f2\ c\ s\ obj$
$\bullet\ (f1\ \fatsemi_f\ f2)\ c\ obj\ s = f2\ c\ (f1\ c\ obj\ s)\ s$

same_to_level
$\vdash \forall\ n\ obj1\ obj2$
$\bullet\ same\_to\_level\ 0$
$= \{(obj1, obj2)$
$|objectClass\ obj1 = objectClass\ obj2\}$
$\land\ same\_to\_level\ (n + 1)$
$= \{(obj1, obj2)$
$|objectContains\ obj1 = objectContains\ obj2$
$\land\ objectClass\ obj1 = objectClass\ obj2$
$\land\ \#\ (objectRefers\ obj1)$
$= \#\ (objectRefers\ obj2)$
$\land\ Elems$
$(Combine$
$(objectRefers\ obj1)$
$(objectRefers\ obj2))$
$\subseteq same\_to\_level\ n\}$

factor_level $\vdash \forall\ n$
$\bullet\ factor\_level\ n$
$= \{factor$
$|\forall\ c\ state\ obj$
$\bullet\ (obj, factor\ c\ obj\ state) \in same\_to\_level\ n\}$

Factorisation
$\vdash \exists\ f\bullet\ TypeDefn\ (\lambda\ x\bullet\ true)\ f$

MkFactorisation
factor0
factor1
factor2
factor3 $\vdash \forall\ t\ x1\ x2\ x3\ x4$
$\bullet\ factor0\ (MkFactorisation\ x1\ x2\ x3\ x4) = x1$
$\wedge\ factor1\ (MkFactorisation\ x1\ x2\ x3\ x4) = x2$
$\wedge\ factor2\ (MkFactorisation\ x1\ x2\ x3\ x4) = x3$
$\wedge\ factor3\ (MkFactorisation\ x1\ x2\ x3\ x4) = x4$
$\wedge\ MkFactorisation$
$(factor0\ t)$
$(factor1\ t)$
$(factor2\ t)$
$(factor3\ t)$
$= t$

composite $\vdash \forall\ f$
$\bullet\ composite\ f$
$= factor0\ f$
$\fatsemi_f\ factor1\ f$
$\fatsemi_f\ factor2\ f$
$\fatsemi_f\ factor3\ f$

factor_out $\vdash \forall\ f\ s\ r$
$\bullet\ factor\_out\ f\ (s, r)$
$= composite\ f\ (reqClearance\ r)\ (reqSsql\ r)\ s$

levelled_factorisation
$\vdash \forall\ facts$
$\bullet\ facts \in levelled\_factorisation$
$\Leftrightarrow factor0\ facts \in factor\_level\ 0$
$\wedge\ factor1\ facts \in factor\_level\ 1$
$\wedge\ factor2\ facts \in factor\_level\ 2$
$\wedge\ factor3\ facts \in factor\_level\ 3$

FactoredMachine
$\vdash \exists\ f\bullet\ TypeDefn\ (\lambda\ x\bullet\ true)\ f$

MkFactoredMachine
machine
factors $\vdash \forall\ t\ x1\ x2$
$\bullet\ machine\ (MkFactoredMachine\ x1\ x2) = x1$
$\wedge\ factors\ (MkFactoredMachine\ x1\ x2) = x2$
$\wedge\ MkFactoredMachine\ (machine\ t)\ (factors\ t) = t$

well_factored
$\vdash \forall\ m$
$\bullet\ m \in well\_factored$
$\Leftrightarrow factors\ m \in levelled\_factorisation$
$\wedge\ Output\ (machine\ m) = factor\_out\ (factors\ m)$

special_machine
$\vdash \forall\ (m, f1, f2)\ rl1\ rl2\ r$
$\bullet\ special\_machine\ (m, f1, f2)\ rl1\ r\ rl2$
$= (let\ s1 = Snd\ (lift\_machine\ m\ (Init\ m, rl1))$

```
                and s2 = Snd (lift_machine m (Init m, rl2))
                in let pe_req
                       = f1 (reqClearance r) (reqSsql r) s1
                   in f2 (reqClearance r) pe_req s2)
same_at_c_below_level
            ⊢ ∀ n c obj1 obj2
              • same_at_c_below_level 0 c = identicalObj c
                ∧ same_at_c_below_level (n + 1) c
                = {(obj1, obj2)
                |# (objectRefers obj1) = # (objectRefers obj2)
                    ∧ Elems
                        (Combine
                            (objectRefers obj1)
                            (objectRefers obj2))
                      ⊆ same_at_c_below_level n c}
label_secure_to
            ⊢ ∀ n
              • label_secure_to n
                = {mff
                |∀ rl r
                  • (let sm = special_machine mff rl r
                     in sm
                       ∈ ml_secure
                         sameRequests
                         (same_at_c_below_level n))}
label_secure  ⊢ label_secure
                = {fm
                |let m = machine fm and fs = factors fm
                    in let lf1 = factor0 fs
                      and lf2 = factor0 fs ⨾f factor1 fs
                      and lf3
                          = factor0 fs
                            ⨾f factor1 fs
                            ⨾f factor2 fs
                      and rf1
                          = factor1 fs
                            ⨾f factor2 fs
                            ⨾f factor3 fs
                      and rf2 = factor2 fs ⨾f factor3 fs
                      and rf3 = factor3 fs
                      in (m, lf1, rf1) ∈ label_secure_to 1
                        ∧ (m, lf2, rf2) ∈ label_secure_to 2
                        ∧ (m, lf3, rf3) ∈ label_secure_to 3}
simplest_witness
            ⊢ simplest_witness
              = (let next sr = Arbitrary
                 and output sr = Arbitrary
                 and init = Arbitrary
```

```
    and f0 c ob s = Arbitrary
    in let (f1, f2, f3) = (f0, f0, f0)
      in let mach = MkMachine next output init
        and facs = MkFactorisation f0 f1 f2 f3
        in MkFactoredMachine mach facs)
```

identity_witness

```
⊢ identity_witness
  = (let next = Fst
    and output = reqSsql o Snd
    and init = Arbitrary
    and f0 c ob s = ob
    in let (f1, f2, f3) = (f0, f0, f0)
      in let mach = MkMachine next output init
        and facs = MkFactorisation f0 f1 f2 f3
        in MkFactoredMachine mach facs)
```

purge_above_level

```
⊢ ConstSpec
  (λ purge_above_level'
    • (∀ c ob
      • purge_above_level' 0 c ob = FilterObj c ob)
      ∧ (∀ n c d t os
      • purge_above_level'
          (n + 1)
          c
          (MkObj (d, t, os))
        = MkObj
          (Arbitrary, Arbitrary,
            Map (purge_above_level' n c) os)))
  purge_above_level
```

5.7 Theorems

purge_above_level_consistent

```
⊢ Consistent
  (λ purge_above_level'
    • (∀ c ob
      • purge_above_level' 0 c ob = FilterObj c ob)
      ∧ (∀ n c d t os
      • purge_above_level'
          (n + 1)
          c
          (MkObj (d, t, os))
        = MkObj
          (Arbitrary, Arbitrary,
            Map (purge_above_level' n c) os)))
```

purge_above_level_thm

```
⊢ ∀ n c ob
  • purge_above_level (n + 1) c ob
```

$= MkTree$
$((Arbitrary,\ Arbitrary),$
Map
$(purge\_above\_level\ n\ c)$
$(objectRefers\ ob))$

same_at_c_below_level_purge_above_level_thm1

$\vdash \forall\ n\ c\ ob1\ ob2$
$\bullet\ (ob1,\ ob2) \in same\_at\_c\_below\_level\ n\ c$
$\Rightarrow purge\_above\_level\ n\ c\ ob1$
$= purge\_above\_level\ n\ c\ ob2$

same_at_c_below_level_purge_above_level_thm2

$\vdash \forall\ n\ c\ ob1\ ob2$
$\bullet\ purge\_above\_level\ n\ c\ ob1$
$= purge\_above\_level\ n\ c\ ob2$
$\Rightarrow (ob1,\ ob2) \in same\_at\_c\_below\_level\ n\ c$

same_at_c_below_level_purge_above_level_thm

$\vdash \forall\ n\ c\ ob1\ ob2$
$\bullet\ (ob1,\ ob2) \in same\_at\_c\_below\_level\ n\ c$
$\Leftrightarrow purge\_above\_level\ n\ c\ ob1$
$= purge\_above\_level\ n\ c\ ob2$

simple_witness_thm

$\vdash machine\ simplest\_witness$
$= MkMachine$
$(\lambda\ sr\bullet\ Arbitrary)$
$(\lambda\ sr\bullet\ Arbitrary)$
$Arbitrary$
$\wedge\ factors\ simplest\_witness$
$= MkFactorisation$
$(\lambda\ c\ ob\ s\bullet\ Arbitrary)$
$(\lambda\ c\ ob\ s\bullet\ Arbitrary)$
$(\lambda\ c\ ob\ s\bullet\ Arbitrary)$
$(\lambda\ c\ ob\ s\bullet\ Arbitrary)$

identity_witness_thm

$\vdash machine\ identity\_witness$
$= MkMachine\ Fst\ (reqSsql\ o\ Snd)\ Arbitrary$
$\wedge\ factors\ identity\_witness$
$= MkFactorisation$
$(\lambda\ c\ ob\ s\bullet\ ob)$
$(\lambda\ c\ ob\ s\bullet\ ob)$
$(\lambda\ c\ ob\ s\bullet\ ob)$
$(\lambda\ c\ ob\ s\bullet\ ob)$

same_at_c_below_level_equiv_thm

$\vdash \forall\ n\ c\bullet\ same\_at\_c\_below\_level\ n\ c \in Equivalence$

lift_rel_same_at_c_below_level_indexed_equiv_thm

$\vdash \forall\ n\bullet\ LiftRel\ (same\_at\_c\_below\_level\ n) \in IndexedEquiv$

lift_rel_same_at_c_below_level_refl_thm

$\vdash \forall\ n\ C\ x\bullet\ (x,\ x) \in LiftRel\ (same\_at\_c\_below\_level\ n)\ C$

simplest_witness_label_secure_thm

$\vdash$ *simplest_witness* $\in$ *label_secure*

identity_witness_label_secure_thm

$\vdash$ *identity_witness* $\in$ *label_secure*

6 INDEX