event-structure-theory[11,40]

ABS: event_system_typename() **event_system_typename**

STM: event_system_typename_wf

ABS: EventsWithOrder **EOrder**

STM: EOrder_wf

ABS: EventsWithState **E-State**

STM: E-State_wf

STM: EState-subtype-EOrder

ABS: EventsWithKinds **EKind**

STM: EKind_wf

ABS: EventsWithValues **EVal**

STM: EVal_wf

ABS: ESMachineAxiom(E;T;V;M;loc;knd;val;$when$;$after$;$sndr$;$Trans$;$Send$;$Choose$) **ESMachineAxiom**

STM: ESMachineAxiom_wf

ABS: $e = e'$ **es-eq-E**

STM: es-eq-E_wf

STM: assert-es-eq-E

STM: assert-es-eq-E-2

STM: decidable__es-E-equal

STM: test-eq-E-update

ABS: es-LnkTag-deq **es-LnkTag-deq**

STM: es-LnkTag-deq_wf

ABS: case(kind(e))act(a) => f(a)rcv(l,tg) => g(l;tg) **es-kindcase**

STM: es-kindcase_wf

ABS: msgtype(m) **es-msgtype**

STM: es-msgtype_wf

STM: es-valtype-kindtype

ABS: state@i\\x **es-state-without**

STM: es-state-without_wf

STM: es-state-eta

STM: event_system-level-subtype

ABS: mk-eval(E;eq;prd;$info$;oax;T;w;a;sax;V;v) **mk-eval**

STM: mk-eval_wf

ABS: AtomFreeDecls(X) **EVal-atom-free**

ABS: state when e\\x **es-state-when-without**

STM: es-state-when-without_wf

ABS: state after e\\x **es-state-after-without**

STM: es-state-after-without_wf

ABS: e c$\leq$ e' **es-causle**

STM: es-causle_wf

STM: es-locl-trans

STM: es-locl-trichotomy

STM: es-le-trans

STM: es-locl_transitivity

STM: es-locl_transitivity1

STM: es-locl_transitivity2

STM: es-le_transitivity

STM: es-le_weakening

STM: es-le_weakening_eq

STM: es-locl-trans-test

STM: es-locl-irreflex-test

STM: es-le-trans2

STM: es-le-trans3

STM: es-index-zero

STM: es-causle-trans

STM: es-causl-trans3

STM: es-causle_transitivity

STM: es-causl_transitivity

STM: es-causl_transitivity1

STM: es-causl_transitivity2

STM: es-causle_weakening

STM: es-causle_weakening_eq

STM: es-causle_weakening_locl

STM: es-causl_weakening

STM: es-causl-trans-test

STM: es-le-causle

STM: es-locl-total

STM: es-locl-total2

STM: same-loc-total

STM: same-loc-total2

STM: es-le-total

STM: es-locl-swellfnd

STM: es-causl-wellfnd

STM: es-causl-swellfnd

STM: es-le-not-locl

STM: es-causal-antireflexive

STM: es-causl_irreflexivity

STM: es-causal-antisymmetric

STM: es-causl-locl

STM: es-causle-le

STM: es-pred-locl

STM: es-le-self

STM: es-pred-le

STM: es-pred-causl

STM: es-sender-causl

STM: es-sender-causle

STM: causl-trans-test2

STM: es-causle-retraction

ABS: $f{**}(e)$ **es-fix**

STM: es-fix_wf

STM: es-fix-cases

STM: es-time-sender

STM: es-first-implies

STM: es-loc-rcv

STM: es-isrcv-loc

STM: es-hasloc

STM: es-loc-sender

STM: same-sender-index

STM: es-le-iff

STM: es-first-le

STM: es-le-antisymmetric

STM: es-le_antisymmetry

STM: es-first-unique

STM: es-causl-iff

STM: implies-es-pred

STM: es-le-pred

STM: implies-es-pred2

STM: es-pred-one-one

STM: decidable__es-locl

STM: es-next

ABS: $e <\text{loc}\ e'$ **es-bless**

STM: es-bless_wf

STM: assert-es-bless

STM: decidable__es-le

ABS: es-ble{i:l}(es;e;e') **es-ble**

STM: es-ble_wf

STM: assert-es-ble

STM: decidable__es-causl

STM: decidable__es-causle

STM: decidable__existse-causl

STM: decidable__existse-causle

STM: decidable__alle-causl

STM: decidable__alle-causle

STM: decidable__rel_plus-causl

STM: decidable__rel_star-causl

STM: causal-sort

STM: es-sends-bound

STM: decidable__existse-rcv

STM: es-minimal-event

ABS: es-bc{i:l}(es;e;e') **es-bc**

STM: es-bc_wf

STM: assert-es-bc

ABS: $\exists e{=}k(v).P(e;v)$ **es-ek**

ABS: $\exists e$:rvc(l,tg,v).$P(e;v)$ **es-er**

ABS: mval(m) **es-mval**

STM: es-mval_wf

STM: es-mval-valtype

STM: es-msg-rcvd

ABS: before(e) **es-before**

STM: es-before_wf

STM: es-before_wf2

STM: member-es-before

STM: l_before-es-before

STM: l_before-es-before-iff

ABS: es-le-before(es;e) **es-le-before**

STM: es-le-before_wf

STM: member-es-le-before

STM: es-causle-list

ABS: es-init(es;e) **es-init**

STM: es-init_wf

STM: es-init-le

STM: es-first-init

STM: es-init-identity

STM: es-init-elim

STM: es-init-elim2

STM: es-init-eq

STM: es-loc-init

ABS: [e, e'] **es-interval**

STM: es-interval_wf

STM: member-es-interval

STM: l_before-es-interval

STM: hd-es-interval

STM: es-interval-non-zero

STM: es-interval-nil

STM: es-interval-is-nil

STM: es-interval-last

STM: es-interval-less

STM: es-interval-less-sq

STM: es-interval-eq

STM: es-interval-eq2

STM: es-interval-length-one-one

STM: es-interval-one-one

STM: es-interval-iseg

STM: es-interval-partition

STM: es-interval-select

STM: es-interval_wf2

STM: es-le-before-partition

STM: es-le-before-partition2

ABS: haslnk(l;e) **es-haslnk**

STM: es-haslnk_wf

STM: assert-es-haslnk

ABS: rcvs(l;before(e')) **es-rcvs**

STM: es-rcvs_wf

STM: member-es-rcvs

ABS: snds(l;before(e)) **es-snds**

STM: es-snds_wf

STM: member-es-snds

ABS: snds(l, before(e,n)) **es-snds-index**

STM: es-snds-index_wf

STM: member-es-snds-index

STM: firstn-before

STM: es-before-decomp

STM: last-es-snds-index

ABS: emsg(e) **es-msg**

STM: es-msg_wf

STM: es-msg_wf2

STM: es-msg-member-sends

ABS: msgs(l;before(e')) **es-msgs**

STM: es-msgs_wf

STM: haslink_wf2

STM: member-es-msgs

STM: es-fifo-nil

STM: es-fifo

STM: es-after-pred

STM: es-after-pred2

STM: decl-state-exists

STM: decl-state-subtype

ABS: @i always.$P(x)$ **alle-at1**

STM: alle-at1_wf

ABS: @i always.$P(x_1;x_2)$ **alle-at2**

STM: alle-at2_wf

STM: alle-at-iff

STM: alle-at-not-first

STM: es-invariant1

STM: es-invariant2

STM: es-constant1

ABS: $\exists e@i.P(e)$ **existse-at**

STM: existse-at_wf

STM: change-lemma

STM: change-lemma2

STM: es-first-exists

STM: change-since-init

ABS: $\exists e{\leq}e'.P(e)$ **existse-le**

STM: existse-le_wf

ABS: $\exists e{<}e'.P(e)$ **existse-before**

STM: existse-before_wf

STM: existse-before-iff

STM: decidable__existse-before

STM: existse-le-iff

STM: decidable__existse-le

ABS: $\forall e'{\geq}e.P(e')$ **alle-ge**

STM: alle-ge_wf

ABS: $\forall e{<}e'.P(e)$ **alle-lt**

STM: alle-lt_wf

STM: alle-lt-iff

STM: decidable__alle-lt

ABS: $\forall e{\leq}e'.P(e)$ **alle-le**

STM: alle-le_wf

STM: alle-le-iff

STM: decidable__alle-le

ABS: $\forall e{\in}[e_1,e_2).P(e)$ **alle-between1**

STM: alle-between1_wf

STM: alle-between1-trivial

STM: alle-between1-true

STM: alle-between1-false

STM: alle-between1_functionality_wrt_iff

STM: decidable__alle-between1

ABS: $\exists e{\in}[e_1,e_2).P(e)$ **existse-between1**

STM: existse-between1_wf

STM: existse-between1-true

STM: existse-between1-false

STM: existse-between1_functionality_wrt_iff

STM: decidable__existse-between1

ABS: $\forall e{\in}[e_1,e_2].P(e)$ **alle-between2**

STM: alle-between2_wf

STM: alle-between2-true

STM: alle-between2-false

STM: alle-between2_functionality_wrt_iff

STM: decidable__alle-between2

ABS: $\exists e{\in}[e_1,e_2].P(e)$ **existse-between2**

STM: existse-between2_wf

STM: existse-between2-false

STM: existse-between2-true

STM: existse-between2_functionality_wrt_iff

STM: decidable__existse-between2

ABS: $\exists e{\in}(e_1,e_2].P(e)$ **existse-between3**

STM: existse-between3_wf

STM: existse-between3-false

STM: existse-between3-true

STM: existse-between3_functionality_wrt_iff

STM: decidable__existse-between3

ABS: $\forall e{\in}(e_1,e_2].P(e)$ **alle-between3**

STM: alle-between3_wf

STM: alle-between3-false

STM: es-subinterval

STM: last-change

ABS: e is first@ i s.t. $e.P(e)$ **es-first-at**

STM: es-first-at_wf

STM: es-first-before

STM: es-first-before2

ABS: es-first-at-since(es;i;e;$e.R(e)$;$e.P(e)$) **es-first-at-since**

STM: es-first-at-since_wf

STM: previous-event-exists

STM: es-first-at-since-iff

ABS: es-first-at-since'(es;i;e;$e.R(e)$;$e.P(e)$) **es-first-at-since'**

STM: es-first-at-since'_wf

ABS: $\forall e{=}\mathrm{rcv}(l,tg).P(e)$ **alle-rcv**

STM: alle-rcv_wf

ABS: $\exists e{=}\mathrm{rcv}(l,tg).P(e)$ **existse-rcv**

STM: existse-rcv_wf

STM: es-bound-list

STM: es-bound-list2

STM: es-machine-axiom

ABS: e receives $\|$ a **es-rcv-atom**

STM: es-rcv-atom_wf

ABS: e sends $\|$ a **es-send-atom**

STM: es-send-atom_wf

ABS: e sends to $i \parallel a$ **es-send-atom-to**

STM: es-send-atom-to_wf

ABS: e leaks x to e' **es-leaks**

ABS: e copies x **es-copies**

STM: state-after-pred

STM: implies-es-atom-axiom

ABS: $i \parallel a$ **es-atom**

STM: es-atom_wf

STM: es-copies_wf

STM: es-leaks_wf

STM: es-atom-axiom

STM: es-atom-lemma1

STM: es-atom-lemma2

ABS: @i discrete ds **es-dds**

STM: es-dds_wf

STM: es-dds-single

ABS: discrete state@i **es-dstate**

STM: es-dstate_wf

STM: es-state-dstate-subtype

STM: es-dstate-subtype

ABS: (discrete state when e) **es-dstate-when**

STM: es-dstate-when_wf

STM: es-state-when-dstate-when

ABS: (discrete state after e) **es-dstate-after**

STM: es-dstate-after_wf

STM: es-state-after-dstate-after

STM: dstate-after-pred

STM: alle-between1-after

STM: alle-between1-after-1

ABS: @i stable $state.P(state)$ **es-stable**

STM: es-stable_wf

STM: stable-implies

STM: stable-implies2

STM: stable-implies3

STM: stable-implies4

ABS: @i Precondition for a(Outcome(p)) P discrete state(ds) **discrete-pre-p**

STM: discrete-pre-p_wf

STM: last-event

ABS: last-solution(es;P;d) **last-solution**

STM: last-solution_wf

STM: last-transition

STM: last-decidable

STM: last-state-change

STM: last-state-change2

STM: last-state-change3

ABS: @i only L affect x:T **es-frame**

STM: es-frame_wf

STM: frame-p-es-frame

STM: es-stable-1

STM: es-stable-2

STM: es-stable-3

STM: es-constant-1

ABS: es-responsive(es;l_1;tg_1;l_2;tg_2) **es-responsive**

STM: es-responsive_wf

STM: es-responsive-bijection

ABS: only $k(v)$:B sends $[tg,f(s;v)]$: T on l **es-only-sender**

STM: es-only-sender_wf

ABS: @i x has type T **es-type**

STM: vartype-es-type

STM: vartype-es-state-sub

STM: es-state-subtype

STM: es-state-subtype-iff

STM: es-state-subtype2

STM: state-after-pred-ds

STM: es-invariant

STM: state-when-first

STM: es-when-first

STM: es-when-init

STM: es-discrete-when-first

STM: es-when-first-discrete

STM: dds-state-after-elapsed

STM: dds-init-elapsed

STM: init-p-implies

ABS: usends1-p(es;ds;k;T;l;tg;B;f) **usends1-p**

STM: usends1-p_wf

STM: usends1-function

ABS: sends1-p(es;x;A;k;B;l;tg;T;f) **sends1-p**

STM: sends1-p_wf

STM: sends-p-implies-sends1-p

ABS: k(v:B) sends f(x:A,v) on l tagged with tg:T provided c(x,v) **conditional-send1-p**

STM: conditional-send1-p_wf

STM: sends1-p-implies-conditional-send1-p

STM: conditional-send1-function

ABS: k(v:B) sends on l [tg:T, f <state, v>]?[] **conditional-send-p**

STM: conditional-send-p_wf

STM: sends-p-implies-conditional-send-p

ABS: pre-init1-p(es;i;x;X;x_0;a;p;P) **pre-init1-p**

STM: pre-init1-p_wf

ABS: weak-precond-send-p(es;T;A;l;tg;a;ds;P;f) **weak-precond-send-p**

STM: weak-precond-send-p_wf

ABS: discrete-weak-precond-send-p(es;T;A;l;tg;a;ds;P;f) **discrete-weak-precond-send-p**

STM: discrete-weak-precond-send-p_wf

STM: wps-implies-discrete-wps

ABS: weak-send-do-apply(es;T;l;tg;a;ds;f) **weak-send-do-apply**

STM: weak-send-do-apply_wf

ABS: @i locl(a) occurs once **once-p**

STM: once-p_wf

ABS: locl(a) sends [tg,$f\{A\rightarrow T\}(x)$] on link l once **send-once-p**

STM: send-once-p_wf

ABS: recognizer-p(es;T;A;P;k;i;r;x) **recognizer-p**

STM: recognizer-p_wf

ABS: recognizer(es;i;ds;x;k;T;$test$) **recognizer**

STM: recognizer_wf

ABS: @i k(v:T) triggers local action a when P (x:A) v **trigger1-p**

STM: trigger1-p_wf

STM: es-interval-induction

STM: es-interval-induction2

ABS: PossibleEvent($poss$) **possible-event**

STM: possible-event_wf

ABS: pe-es(e) **pe-es**

STM: pe-es_wf

ABS: pe-e(p) **pe-e**

STM: pe-e_wf

ABS: pe-state(p) **pe-state**

STM: pe-state_wf

ABS: pe-loc(p) **pe-loc**

STM: pe-loc_wf

ABS: K(P)@e **es-knows**

STM: es-knows_wf

STM: es-knows-true

STM: es-knows-knows

STM: es-knows-not

STM: es-knows-trans

STM: es-knows-valid

ABS: $e_1 \leq e_2$ **poss-le**

STM: poss-le_wf

STM: es-knows-stable

ABS: $e{:}s.P(s)@j$ **es-simul**

STM: es-simul_wf

ABS: es-decls(es;i;ds;da) **es-decls**

STM: es-decls_wf

STM: es-decls-iff

STM: es-decls-join-single

ABS: with decls ds dasends on l from e include $f(e)$ and only these for tags in tgs

es-sends-iff

STM: es-sends-iff_wf

ABS: state dsk:A sends [tg, $e.f(e)$:B] on l **es-kind-sends-iff**

STM: es-kind-sends-iff_wf

STM: es-sends-iff_functionality

ABS: es-update-iff(es;i;x;ds;$e.P(e)$;$s.f(s)$) **es-update-iff**

STM: es-update-iff_wf

ABS: (e sends on l with tag tg) **es-sends-on**

STM: es-sends-on_wf

ABS: es-first-from(es;e;l;tg) **es-first-from**

STM: es-first-from_wf

STM: es-kind-first-from

STM: es-loc-first-from

ABS: loc-on-path(es;i;L) **loc-on-path**

STM: loc-on-path_wf

STM: loc-on-path-append

STM: loc-on-path-cons

STM: loc-on-path-nil

STM: loc-on-path-singleton

STM: es-sender-first-from

STM: es-first-from-is-first

ABS: es-sends-iff2(es;l;tg;B;ds;$e.P(e)$;$e.f(e)$) **es-sends-iff2**

STM: es-sends-iff2_wf

STM: es-sends-iff2_functionality

ABS: event-info(ds;da) **event-info**

STM: event-info_wf

ABS: es-info(es;e) **es-info**

STM: es-info_wf

ABS: es-hist{i:l}(es;e_1;e_2) **es-hist**

STM: es-hist_wf

STM: member-es-hist

STM: null-es-hist

STM: es-hist-iseg

STM: es-hist-partition

STM: es-hist-last

STM: last-es-hist

STM: es-hist-is-append

STM: es-hist-is-concat

STM: iseg-es-hist

STM: es-hist-one-one

ABS: es-trans-state-from{i:l}(es;ks;g;z;e_1;e_2) **es-trans-state-from**

STM: es-trans-state-from_wf

ABS: e_2 = first $e \geq e_1.P(e)$ **es-first-since**

STM: es-first-since_wf

STM: es-first-since_functionality_wrt_iff

STM: alle-between1-not-first-since

STM: alle-between2-not-first-since

STM: es-increasing-sequence

STM: es-increasing-sequence2

ABS: $[e_1;e_2]\sim([a,b].p(a;b))*[a,b].q(a;b)$ **es-pstar-q**

STM: es-pstar-q_wf

STM: es-pstar-q-trivial

STM: es-pstar-q-le

STM: es-pstar-q_functionality_wrt_implies

STM: es-pstar-q_functionality_wrt_rev_implies

STM: es-pstar-q_functionality_wrt_iff

STM: es-pstar-q-partition

ABS: $[e_1,e_2]\sim([a,b].p(a;b))+$ **es-pplus**

STM: es-pplus_wf

STM: es-pplus_functionality_wrt_implies

STM: es-pplus_functionality_wrt_rev_implies

STM: es-pplus_functionality_wrt_iff

STM: es-pplus-trivial

STM: es-pplus-le

STM: es-pplus-alle-between2

STM: es-pplus-partition

STM: es-pplus-first-since

STM: es-pplus-first-since-exit

ABS: data(T) **data**

STM: data_wf

ABS: secret-table(T) **secret-table**

STM: secret-table_wf

ABS: $\|tab\|$ **st-length**

STM: st-length_wf

ABS: ptr(tab) **st-ptr**

STM: st-ptr_wf

ABS: st-atom(tab;n) **st-atom**

STM: st-atom_wf

ABS: atoms-distinct(tab) **st-atoms-distinct**

STM: st-atoms-distinct_wf

ABS: next(tab) **st-next**

STM: st-next_wf

ABS: key(tab;n) **st-key**

STM: st-key_wf

ABS: data(tab;n) **st-data**

STM: st-data_wf

STM: st-ptr-wf2

ABS: st-lookup(tab;x) **st-lookup**

STM: st-lookup_wf

STM: st-lookup-property

STM: st-lookup-outl

STM: st-lookup-distinct

ABS: st-key-match(tab;k_1;k_2) **st-key-match**

STM: st-key-match_wf

ABS: decrypt(tab;$kval$) **st-decrypt**

STM: st-decrypt_wf

ABS: encrypt(tab;$keyv$) **st-encrypt**

STM: st-encrypt_wf

STM: st-length-encrypt

STM: st-atom-encrypt

ABS: ?[x] **cond-to-list**

STM: cond-to-list_wf

ABS: es-secret-server{$table:ut2, $encrypt:ut2, $decrypt:ut2}
(es; T; L; i)

es-secret-server

STM: es-secret-server_wf

STM: ss-ptr-non-decreasing

STM: ss-table-length

STM: ss-atom-constant

STM: ss-atoms-distinct

STM: ss-encrypt-unique

ABS: es-seq(es;S) **es-seq**

STM: es-seq_wf

STM: send-minimal-lemma

ABS: @e(x→v) **es-change-to**

STM: es-change-to_wf

STM: change-to-lemma

STM: change-to-lemma2

ABS: change-to(x;e) **change-to**

STM: change-to_wf

ABS: x changed before e **changed**

STM: changed_wf

ABS: (last change to x before e) **last-change**

STM: last-change_wf

STM: last-change-property

STM: last-change-after-property

STM: change-to-last-change

STM: after-last-change

STM: loc-last-change

STM: assert-changed

STM: not-changed

STM: has-changed

STM: last-change-pred

STM: init-changed

STM: init-not-changed

STM: last-change-equal

STM: last-change-equal2

STM: es-le-last-change

ABS: lastchange(x;e) **es-lc**

STM: es-lc_wf

STM: es-lc-init-p

STM: es-lc-after

STM: es-loc-lc

STM: es-lc-le

STM: es-after-lc-before

STM: es-lc-no-change

STM: es-lc-no-change2

STM: es-lc-cases

STM: es-lc-cases2

STM: es-lc-bool

STM: es-lc-btrue

STM: es-lc-unique

STM: es-lc-equal

STM: once-lemma

STM: const-lemma1

STM: const-lemma

STM: next-state-relation

STM: next-var-value

ABS: next event in [e;*bound*] after which $x = v$ **es-next-assign**

STM: es-next-assign_wf

STM: es-next-assign-property

STM: es-next-assign-unique

ABS: next event in [e,*bound*] after which $x =_b b$ **es-next-bool-assign**

STM: es-next-bool-assign_wf

STM: es-next-bool-assign-property

ABS: (x unchanged-for t @ e) **unchanged-for**

STM: unchanged-for_wf

STM: es-causle-time

STM: es-causl-fifo

STM: unchanged-for-change-to

ABS: val(e) $\equiv$ val(e') **es-same-val**

STM: es-same-val_wf

ABS: AbsInterface(A) **es-interface**

STM: es-interface_wf

STM: es-interface-subtype_rel

ABS: *f'Ia* **es-interface-image**

STM: es-interface-image_wf

STM: p-first_wf-interface

ABS: es-in-port(es;l;tg) **es-in-port**

STM: es-in-port_wf

ABS: es-trigger(es;i;knd;ds;f) **es-trigger**

STM: es-trigger_wf

ABS: es-triggers(es;i;ds;$conds$) **es-triggers**

STM: es-triggers_wf

ABS: es-in-port-conds(A;l;tg) **es-in-port-conds**

STM: es-in-port-conds_wf

STM: es-in-port-triggers

ABS: es-triggers-params-consistent(es;A;i;ds;$conds$) **es-triggers-params-consistent**

STM: es-triggers-params-consistent_wf

STM: es-triggers-params-join

STM: es-triggers-params-list-join

STM: functions-decl-state

STM: es-interface-conditional

ABS: es-interface-left(X) **es-interface-left**

STM: es-interface-left_wf

ABS: es-interface-right(X) **es-interface-right**

STM: es-interface-right_wf

ABS: $e \in_b X$ **es-is-interface**

STM: es-is-interface_wf

ABS: es-interface-empty(es;I) **es-interface-empty**

STM: es-interface-empty_wf

ABS: Empty **es-empty-interface**

STM: es-empty-interface_wf

ABS: isempty{isempty_compseq_tag_def:ObjectId}(e) **isempty_compseq_tag_def**

STM: es-empty-interface-property

STM: es-trigger-not-loc

STM: es-trigger-loc

STM: es-triggers-not-loc

STM: es-triggers-loc

ABS: $\{I\}$ **es-interface-predicate**

STM: es-interface-predicate_wf

STM: es-interface-conditional-domain

STM: es-interface-conditional-domain-iff

STM: es-interface-conditional-predicate-equivalent

STM: es-interface-conditional-domain-member

ABS: E(X) **es-E-interface**

STM: es-E-interface_wf

STM: decidable__equal_es-E-interface

ABS: es-interface-sublist(X;z) **es-interface-sublist**

STM: es-interface-sublist_wf

STM: es-E-interface-subtype

STM: es-E-interface-subtype_rel

STM: es-E-interface-strong-subtype

STM: es-E-interfaces-strong-subtype

STM: es-E-interface_functionality

STM: es-E-interface-conditional

STM: es-E-interface-conditional2

STM: es-E-interface-conditional-subtype1

STM: es-E-interface-conditional-subtype2

STM: es-causle-interface-retraction

STM: es-fix_wf2

STM: es-fix_property

STM: es-fix-equal

STM: es-fix-step

STM: es-fix-connected

STM: es-fix-sqequal

STM: es-fix-equal-E-interface

STM: fun-connected-causle

STM: loc-on-path-decomp

STM: es-fix-causle

STM: es-fix-causl

STM: es-is-interface-image

STM: es-E-interface-image

STM: es-E-interface-predicate

ABS: es_E_interface_predicate{es_E_interface_predicate_compseq_tag_def:ObjectId}
(I; es)

es_E_interface_predicate_compseq_tag_def

ABS: $X(e)$ **es-interface-val**

STM: es-interface-val_wf

STM: es-interface-val_wf2

ABS: $X(L)$ **es-interface-vals**

STM: es-interface-vals_wf

STM: es-interface-vals-append

STM: es-interface-image-val

STM: es-interface-extensionality

STM: es-interface-image-trivial

STM: es-interface-val-conditional

STM: es-interface-val-disjoint

ABS: $(I|p)$ **es-interface-restrict**

STM: es-interface-restrict_wf

ABS: $(I|\neg p)$ **es-interface-co-restrict**

STM: es-interface-co-restrict_wf

STM: es-is-interface-restrict

STM: es-is-interface-restrict-guard

STM: es-is-interface-restrict2

STM: es-is-interface-co-restrict

STM: es-interface-val-restrict

STM: es-interface-val-restrict-sq

STM: es-interface-val-co-restrict

STM: es-interface-restrict-trivial

STM: es-interface-restrict-idempotent

STM: es-E-interface-restrict

STM: es-E-interface-co-restrict

ABS: $X \cap Y = 0$ **es-interface-disjoint**

STM: es-interface-disjoint_wf

STM: es-interface-restrict-disjoint

STM: es-interface-restrict-conditional

ABS: $X|a.P(a)$ **es-interface-filter**

STM: es-interface-filter_wf

STM: es-is-interface-filter

STM: es-interface-filter-val

STM: es-is-interface-in-port

STM: es-is-interface-trigger

STM: es-is-interface-triggers

STM: es-is-interface-triggers2

STM: es-triggers-conditional

STM: es-is-interface-p-first

STM: es-E-interface-p-first

STM: es-triggers-p-first

STM: es-in-port-val

STM: es-trigger-val

STM: es-triggers-val

STM: es-in-port-receives

ABS: X is local **es-interface-local**

STM: es-interface-local_wf

ABS: X is finite **es-interface-finite**

STM: es-interface-finite_wf

STM: es-interface-finite-implies

ABS: es-interface-history(es;X;e) **es-interface-history**

STM: es-interface-history_wf

STM: es-interface-history-first

STM: es-interface-history-pred

STM: es-interface-history-iseg

STM: member-es-interface-history

STM: nonempty-es-interface-history

STM: es-interface-from-decidable

ABS: es-p-local-pred(es;P) **es-p-local-pred**

STM: es-p-local-pred_wf

ABS: es-p-le-pred(es;P) **es-p-le-pred**

STM: es-p-le-pred_wf

STM: decidable__es-p-local-pred

STM: decidable__es-p-le-pred

STM: decidable__exists-es-p-local-pred

STM: decidable__exists-es-p-le-pred

STM: es-interface-local-pred

STM: es-interface-le-pred

STM: es-interface-local-pred-bool

STM: es-interface-le-pred-bool

ABS: last(P) **es-local-pred**

STM: es-local-pred_wf

STM: es-local-pred-property

ABS: es-local-le-pred{i:l}(es;P) **es-local-le-pred**

STM: es-local-le-pred_wf

STM: es-local-le-pred-property

ABS: prior(X) **es-prior-interface**

STM: es-prior-interface_wf

ABS: le(X) **es-le-interface**

STM: es-le-interface_wf

STM: es-is-prior-interface

STM: es-is-prior-interface-pred

STM: es-is-le-interface

STM: es-is-le-interface-iff

STM: es-prior-interface-val

STM: es-le-interface-val

STM: es-le-interface-val-cases

STM: es-prior-interface-val-pred

STM: es-prior-interface-locl

STM: es-prior-interface-causl

STM: es-le-prior-interface-val

STM: es-prior-interface-val-locl

STM: es-le-interface-le

STM: es-le-interface-causle

STM: es-interface-history-prior

ABS: prior-state(f;$base$;X;e) **es-local-prior-state**

STM: es-local-prior-state_wf

ABS: local-state(f;$base$;X;e) **es-interface-local-state**

STM: es-interface-local-state_wf

STM: es-interface-local-state-prior

STM: es-interface-local-state-cases

ABS: es-prior-interface-vals(es;X;e) **es-prior-interface-vals**

STM: es-prior-interface-vals_wf

STM: es-prior-interface-vals-property

STM: local-prior-state-accumulate

ABS: $\Sigma \leq e(X)$ **es-interface-sum**

STM: es-interface-sum_wf

STM: es-interface-sum-cases

STM: es-interface-sum-le-interface

ABS: prior-f-fixedpoints(e) **es-prior-fixedpoints**

STM: es-prior-fixedpoints_wf

STM: es-prior-fixedpoints-fix

STM: member-es-fix-prior-fixedpoints

STM: es-fix-last-prior-fixedpoints

STM: es-prior-fixedpoints-non-null

STM: es-prior-fixedpoints-causle

STM: es-prior-fixedpoints-iseg

STM: es-prior-fixedpoints-no_repeats

STM: es-prior-fixedpoints-fixed

STM: es-prior-fixedpoints-unequal

ABS: ComponentSpec(A;B) **es-component**

STM: es-component_wf

ABS: es-decl(es;ds;da) **es-decl**

STM: es-decl_wf

ABS: $[P?\ f : g]$ **conditional**

STM: conditional_wf

STM: conditional_wf2

STM: conditional-idempotent

STM: conditional-ifthenelse

STM: conditional-apply

STM: conditional_wf-interface

STM: conditional_wf-interface2

ABS: $Q \leftarrow\!=\!=f\!=\!=\ P$ **weak-antecedent-function**

STM: weak-antecedent-function_wf

STM: weak-antecedent-function-property

STM: weak-antecedent-function_functionality_wrt_pred_equiv

STM: weak-antecedent-functions-compose

STM: weak-antecedent-conditional

ABS: $Q \leftarrow\!\leftarrow\!=\ f\!=\!=\ P$ **weak-antecedent-surjection**

STM: weak-antecedent-surjection_wf

STM: weak-antecedent-surjection-property

STM: weak-antecedent-surjection_functionality_wrt_pred_equiv

STM: weak-antecedent-surjections-compose

STM: weak-antecedent-surjection-conditional

STM: weak-antecedent-surjection-conditional2

ABS: f is Q-R-pre-preserving on P **Q-R-pre-preserving**

STM: Q-R-pre-preserving_wf

STM: Q-R-pre-preserving_functionality_wrt_implies

STM: Q-R-pre-preserving-rewrite-test

STM: Q-R-pre-preserving-compose

STM: Q-R-pre-preserving-1-1

STM: Q-R-pre-preserving-conditional

STM: sender-le-pre-preserving

ABS: f is R-pre-preserving on P **rel-pre-preserving**

STM: rel-pre-preserving_wf

STM: rel-pre-preserving-compose

ABS: f is locl-pre-preserving on P **locl-pre-preserving**

STM: locl-pre-preserving_wf

STM: locl-pre-preserving-compose

STM: locl-pre-preserving-1-1

ABS: g glues $Ia{:}Qa$ $--f\rightarrow$ $Ib{:}Rb$ **Q-R-glues**

STM: Q-R-glues_wf

STM: Q-R-glues-property

STM: Q-R-glues_functionality

STM: Q-R-glues-empty

STM: Q-Q-glues-to-self-image

STM: Q-Q-glues-to-self

STM: inject-composes

STM: Q-R-glues-composes

STM: Q-R-glues-composes2

STM: Q-R-glues-conditional

STM: Q-R-glues-conditional2

STM: Q-R-glues-split

STM: Q-R-glues-trivial-restrict

STM: Q-R-glues-trivial-split

ABS: $Ia{:}Qa$ $\rightarrow -f\rightarrow$ $Ib{:}Rb$ **Q-R-glued**

STM: Q-R-glued_wf

STM: Q-R-glued-empty

STM: Q-Q-glued-self-image

STM: Q-Q-glued-to-self

STM: Q-R-glued-composes

STM: Q-R-glued-conditional

STM: Q-R-glued-first

ABS: g glues Ia $--f\rightarrow$ Ib **glues**

STM: glues_wf

STM: glues-property

ABS: glued(es;B;f;Ia;Ib) **glued**

STM: glued_wf

STM: glued-Q-R-glued

STM: glued-to-self

STM: glue-composes

STM: glued-composes

STM: glued-composes-simple

STM: glued-first

STM: sender-glues-trigger

STM: sender-glues-triggers

STM: sender-glues-triggers2

STM: sender-frame-glues-triggers

ABS: sender-glues-triggers-p(es;A;l;tg;ds;$conds$) **sender-glues-triggers-p**

STM: sender-glues-triggers-p_wf

ABS: triggers-glued-p(es;A;l;tg;ds;$conds$) **triggers-glued-p**

STM: triggers-glued-p_wf

STM: sender-glues-implies-triggers-glued

STM: causal-p-predecessor

ABS: retrace(es;Q;X) **retrace**

STM: retrace_wf

ABS: retracer(p) **retracer**

STM: retracer_wf

ABS: state-machine-spec{i:l}(es;C;R;F;I;O) **state-machine-spec**

STM: state-machine-spec_wf

DIR: abstract chain replication

DIR: chain replication

http://www.nuprl.org/FDLcontent/p0_963683_/p26_61797_{event-structure-theory}.html