The Formal Definition of SWIN Language

Chenglong Wang

May 31, 2014

1 Safety Update Calculus

1.1 Syntax

$$
\begin{array}{lcll}
\Pi & ::= & \{\bar{\pi}\} & \\
\pi & ::= & (\bar{\mathtt{d}})\ [\mathtt{l} : \mathtt{C_l}\ \rightarrow\ \mathtt{r} : \mathtt{C_r}] & \\
\mathtt{d} & ::= & \mathtt{x} : \mathtt{C_1} \hookrightarrow \mathtt{C_2} & (\mathtt{variable}) \\
\mathtt{l} & ::= & \mathtt{new\ C}(\bar{\mathtt{x}}) \mid \mathtt{x.m}(\bar{\mathtt{x}}) & \\
\mathtt{r} & ::= & \mathtt{t} & (\mathtt{FJ\ term})
\end{array}
$$

Variables in $\mathtt{r}$ are meta-variables bounded by the variable definition in π.

1.2 Auxiliary Definition

$$\frac{}{\mathtt{TypeMapping}(\{\bar{\pi}\}) = \bigcup_{\pi}(\mathtt{TypeMapping}(\pi))} \quad \text{(TYPEMAPPING1)}$$

$$\frac{}{\mathtt{TypeMapping}([(\overline{\mathtt{x} : \mathtt{C_1} \hookrightarrow \mathtt{C_2}})\ [\mathtt{l} : \mathtt{C_l}\ \rightarrow\ \mathtt{r} : \mathtt{C_r}]]) = \{\mathtt{C_l} \hookrightarrow \mathtt{C_r}\} \cup \{\overline{\mathtt{C_1} \hookrightarrow \mathtt{C_2}}\}} \quad \text{(TYPEMAPPING2)}$$

$$\frac{\mathtt{C}\ \{\mathtt{K}, \bar{\mathtt{M}}\} \in \mathtt{API} \qquad \mathtt{m_1} : (\bar{\mathtt{C}}_\mathtt{s}) \rightarrow \mathtt{C_d} \in \bar{\mathtt{M}}}{\mathtt{mtype}(\mathtt{m_1}, \mathtt{C}, \mathtt{API}) = (\bar{\mathtt{C}}_\mathtt{s}) \rightarrow \mathtt{C_d}} \quad \text{(MTYPE)}$$

1.3 Evaluation-Π

$$\frac{\mathtt{CL} = \mathtt{class}\ \mathtt{C_1}\ \mathtt{extends}\ \mathtt{C_2}\ \{\ \bar{\mathtt{C}}_\mathtt{i}\ \bar{\mathtt{f}}_\mathtt{i};\ \mathtt{K}\ \bar{\mathtt{M}}\ \}}{\Pi(\mathtt{CL}) = \mathtt{class}\ \Pi(\mathtt{C_1})\ \mathtt{extends}\ \Pi(\mathtt{C_2})\ \{\ \Pi(\bar{\mathtt{C}}_\mathtt{i})\ \bar{\mathtt{f}}_\mathtt{i};\ \Pi(\mathtt{K})\ \overline{\Pi(\mathtt{M})}\ \}} \quad \text{(E-DECLARATION)}$$

$$\frac{\mathtt{K} = \mathtt{C_1}\ (\bar{\mathtt{C}}_2\ \bar{\mathtt{f}}_2)\ \{\mathtt{super}(\bar{\mathtt{f}}_3);\ \mathtt{this}.\bar{\mathtt{f}}_\mathtt{i} = \bar{\mathtt{f}}_\mathtt{j}\}}{\Pi(\mathtt{K}) = \Pi(\mathtt{C_1})\ (\Pi(\bar{\mathtt{C}}_2)\ \bar{\mathtt{f}}_2)\ \{\mathtt{super}(\bar{\mathtt{f}}_3);\ \mathtt{this}.\bar{\mathtt{f}}_\mathtt{i} = \bar{\mathtt{f}}_\mathtt{j}\}} \quad \text{(E-CONSTRUCTOR)}$$

$$\frac{\mathtt{M} = \mathtt{C}_1\ \mathtt{m}(\bar{\mathtt{C}}_\mathtt{m}\ \bar{\mathtt{x}})\ \{\mathtt{return}\ \mathtt{t};\}}{\Pi(\mathtt{M}) = \Pi(\mathtt{C}_1)\ \mathtt{m}(\Pi(\bar{\mathtt{C}}_\mathtt{m})\ \bar{\mathtt{x}})\ \{\mathtt{return}\ \Pi(\mathtt{t});\}}\ \text{(E-METHOD)}$$

$$\frac{\mathtt{C}_0 \hookrightarrow \mathtt{C}_1 \in \mathtt{TypeMapping}(\Pi)}{\Pi(\mathtt{C}_0) = \mathtt{C}_1}\ \text{(E-CLASS)}$$

$$\frac{}{\Pi(\mathtt{t.f}) = \Pi(\mathtt{t}).\mathtt{f}}\ \text{(E-T-FIELD)}$$

$$\frac{}{\Pi((\mathtt{C})\ \mathtt{t}) = (\Pi(\mathtt{C}))\ \Pi(\mathtt{t})}\ \text{(E-T-CAST)}$$

$$\frac{}{\Pi(\mathtt{x}) = \mathtt{x}}\ \text{(E-T-VALUE)}$$

$$\frac{\begin{array}{c}[(\bar{\mathtt{d}})\ \mathtt{new}\ \mathtt{C}_0(\ \bar{\mathtt{x}}\ ) : \mathtt{C}_1\ \ \rightarrow\ \ \mathtt{r} : \mathtt{C}_\mathtt{r}] \in \Pi \\ \overline{\mathtt{x} : \mathtt{C}_1 \hookrightarrow \mathtt{C}_2} \in \bar{\mathtt{d}} \qquad \mathtt{Type}\bar{\mathtt{t}}_\mathtt{u} <: \bar{\mathtt{C}}_1\end{array}}{\Pi(\mathtt{new}\ \mathtt{C}_0(\bar{\mathtt{t}}_\mathtt{u})) = [\bar{\mathtt{x}} \rightarrow \Pi(\mathtt{t}_\mathtt{u})](\mathtt{r})}\ \text{(E-T-NEW)}$$

$$\frac{\begin{array}{c}[(\bar{\mathtt{d}})\ \mathtt{x}_0.\mathtt{m}_0(\ \bar{\mathtt{y}}\ ) : \mathtt{C}_1\ \ \rightarrow\ \ \mathtt{r} : \mathtt{C}_\mathtt{r}] \in \Pi \\ \{\overline{\mathtt{y} : \mathtt{C}_1 \hookrightarrow \mathtt{C}_2}, \mathtt{x}_0 : \mathtt{Typet}_0 \hookrightarrow \mathtt{C}'_\mathtt{x}\} \subseteq \bar{\mathtt{d}} \qquad \mathtt{Type}\bar{\mathtt{t}}_\mathtt{u} <: \bar{\mathtt{C}}_1\end{array}}{\Pi(\mathtt{t}_0.\mathtt{m}_0(\bar{\mathtt{t}}_\mathtt{u})) = [\mathtt{x}_0 \rightarrow \mathtt{t}_0, \bar{\mathtt{y}} \rightarrow \overline{\Pi(\mathtt{t}_\mathtt{u})}](\mathtt{r})}\ \text{(E-T-INVOKE)}$$

$$\frac{\mathtt{no\ other\ inference\ rule\ can\ be\ applied}}{\Pi(\mathtt{new}\ \mathtt{C}_0(\bar{\mathtt{t}}_\mathtt{u})) = \mathtt{new}\ \mathtt{C}_0(\Pi(\mathtt{t}_\mathtt{u}))}\ \text{(E-ALTER-NEW)}$$

$$\frac{\mathtt{no\ other\ inference\ rule\ can\ be\ applied}}{\Pi(\mathtt{t}_0.\mathtt{m}_0(\bar{\mathtt{t}}_\mathtt{u})) = \Pi(\mathtt{t}_0).\mathtt{m}(\overline{\Pi(\mathtt{t}_\mathtt{u})})}\ \text{(E-ALTER-INVOKE)}$$

1.4 Typing Rules for Safety Update Calculus

$$\frac{\mathtt{mtype}(\mathtt{m}, \mathtt{C}_\mathtt{x}, \mathtt{API}_\mathtt{s}) = \bar{\mathtt{C}}_\mathtt{s} \rightarrow \mathtt{C}_\mathtt{d} \qquad \mathtt{E} \vdash \mathtt{x} : \mathtt{C}_\mathtt{x} \hookrightarrow \mathtt{C}'_\mathtt{x}, \bar{\mathtt{y}} : \bar{\mathtt{C}}_\mathtt{y} \hookrightarrow \mathtt{C}'_\mathtt{y} \qquad \bar{\mathtt{C}}_\mathtt{y} <: \bar{\mathtt{C}}_\mathtt{s}}{\mathtt{E} \vdash \mathtt{x.m}(\bar{\mathtt{y}}) : \mathtt{C}_\mathtt{d}} \qquad \text{(T-L1)}$$

$$\frac{\mathtt{C}_1\ \{\mathtt{K}_1, \bar{\mathtt{M}}\} \in \mathtt{API}_\mathtt{s} \qquad \mathtt{K}_1 =: \bar{\mathtt{C}}_\mathtt{s} \rightarrow \mathtt{C}_1 \qquad \mathtt{E} \vdash \bar{\mathtt{x}} : \overline{\mathtt{C}_\mathtt{x} \hookrightarrow \mathtt{C}'_\mathtt{x}} \qquad \bar{\mathtt{C}}_\mathtt{x} <: \bar{\mathtt{C}}_\mathtt{s}}{\mathtt{E} \vdash \mathtt{new}\ \mathtt{C}_1(\bar{\mathtt{x}}) : \mathtt{C}_1} \qquad \text{(T-L2)}$$

$$\frac{\mathtt{E} = \mathtt{E}_1, \mathtt{x} : \mathtt{C} \hookrightarrow \mathtt{D}}{\mathtt{E} \vdash \mathtt{x} : \mathtt{C} \hookrightarrow \mathtt{D}} \qquad \text{(T-VAR)}$$

$$\frac{\mathtt{E} \vdash \bar{\mathtt{x}} : \overline{\mathtt{C} \hookrightarrow \mathtt{D}} \qquad \{\mathtt{API_d}, \bar{\mathtt{x}} : \bar{\mathtt{D}}\} \vdash_{\mathtt{FJ}} \mathtt{t} : \mathtt{C_d}}{\mathtt{E} \vdash \mathtt{t} : \mathtt{C_d}} \qquad \text{(T-R)}$$

$$\frac{\{\bar{\mathtt{x}} : \overline{\mathtt{C} \hookrightarrow \mathtt{D}}\} \vdash \mathtt{l} : \mathtt{C_1},\ \mathtt{r} : \mathtt{C_2}}{[\{\bar{\mathtt{x}} : \overline{\mathtt{C} \hookrightarrow \mathtt{D}}\}\ \mathtt{l} : \mathtt{C_1} \rightarrow \mathtt{r} : \mathtt{C_2}] : \mathtt{C_1} \rightsquigarrow \mathtt{C_2}} \qquad \text{(T-}\pi\text{)}$$

$$\frac{\begin{array}{c}\forall \mathtt{C_1}\ \{\mathtt{K_1}, \bar{\mathtt{M}}\} \in \mathtt{API_s}, \exists \mathtt{C_1} \hookrightarrow \mathtt{D_1} \in \mathtt{TypeMapping}(\Pi) \wedge \mathtt{D_1} \in \mathtt{API_d} \\ \forall \mathtt{C_1}\ \{\mathtt{new}\ \mathtt{C_1}(\bar{\mathtt{C}}_\mathtt{x}\ \bar{\mathtt{x}}), \overline{\mathtt{C_m}\ \mathtt{m}\ (\mathtt{C_y}\ \bar{\mathtt{y}})\{...\}} \in \mathtt{API_s} \\ \exists(\overline{\mathtt{x} : \mathtt{C_x} \hookrightarrow \mathtt{C'_x}})[\mathtt{new}\ \mathtt{C_1}(\bar{\mathtt{x}}) \rightarrow \mathtt{r} : \mathtt{C_r}] \in \Pi, (\mathtt{z} : \mathtt{C_1} \hookrightarrow \mathtt{C'_1}, \overline{\mathtt{y} : \mathtt{C_y} \hookrightarrow \mathtt{C'_y}})[\mathtt{x.m}(\bar{\mathtt{y}}) : \mathtt{C_m} \rightarrow \mathtt{rs} : \mathtt{C_{rs}}] \\ \forall \pi_\mathtt{i}, \pi_\mathtt{j} \in \{\bar{\pi}\}, \mathtt{E} \vdash \pi_\mathtt{i} : \mathtt{C_i} \rightsquigarrow \mathtt{D_i}, \pi_\mathtt{j} : \mathtt{C_j} \rightsquigarrow \mathtt{D_j}, \mathtt{C_i} = \mathtt{C_j} \Rightarrow \mathtt{D_i} = \mathtt{D_j}, \mathtt{C_i} <: \mathtt{C_j} \Rightarrow \mathtt{D_i} <: \mathtt{D_j}\end{array}}{\mathtt{E} \vdash \{\bar{\pi}\} : \bigcup_{\pi : \mathtt{C} \rightsquigarrow \mathtt{D} \in \{\bar{\pi}\}} \{\mathtt{C} \rightsquigarrow \mathtt{D}\}} \qquad \text{(T-}\Pi\text{)}$$

2 Theorem

2.1 Environment

Definition Γ is the environment of a term $\mathtt{t}$. $\Gamma_\mathtt{o} = \bar{\mathtt{x}} : \bar{\mathtt{C}}$, which defines types of all variables in the term. $\Gamma_\mathtt{n}$ defines the environment of term after the application of π on a java client code and $\Gamma_\mathtt{o}$ is defined as the variable environment before adaption.

2.2 Lemma 1

Suppose $\Gamma_\mathtt{o} = \bar{\mathtt{x}} : \bar{\mathtt{C}}$, then $\Gamma_\mathtt{n} = \Pi(\Gamma_\mathtt{o}) = \bar{\mathtt{x}} : \overline{\Pi(\mathtt{C})}$ for any type environment.

Proof: Note that all variables are bounded by the definition of a method $\mathtt{M}$. We assume the the variable type environment $\Gamma_\mathtt{o}$ is for term $\mathtt{t}$. And $\mathtt{t}$ is defined in the body of method $\mathtt{M}$ whose definition is $\mathtt{M} = \mathtt{C_1}\ \mathtt{m}(\bar{\mathtt{C}}_\mathtt{m}\ \bar{\mathtt{x}})\{\mathtt{return}\ \mathtt{t};\}$, then $\Gamma_\mathtt{o} = \bar{\mathtt{x}} : \bar{\mathtt{C}}_\mathtt{m}$. According to the rule E-METHOD, we have $\Pi(\mathtt{M}) = \Pi(\mathtt{C_1})\ \mathtt{m}(\Pi(\bar{\mathtt{C}}_\mathtt{m})\ \bar{\mathtt{x}})\ \{\mathtt{return}\ \Pi(\mathtt{t});\}$. Then all the types of all variables in $\mathtt{t}$ will be $\bar{\mathtt{x}} : \overline{\Pi(\mathtt{C})}$. Thus $\Gamma_\mathtt{n} = \bar{\mathtt{x}} : \overline{\Pi(\mathtt{C})}$. $\square$

2.3 Lemma 2

Suppose Π is well typed under SWIN type system and transform from old client using $\mathtt{API_s}$ to new client code using $\mathtt{API_d}$, then:

$\mathtt{C_1} <: \mathtt{C_2}$ in old client code $\Rightarrow \Pi(\mathtt{C_1}) <: \Pi(\mathtt{C_2})$ in new client code.

Proof: Consider the two possibilities of $\mathtt{C_1}$:

Case-1: $\mathtt{C_1}$ is defined in client code.

In this case, according to the rule E-DECLARATION, we have $\Pi(\mathtt{CL}) = \mathtt{class}\ \Pi(\mathtt{C_1})\ \mathtt{extends}\ \Pi(\mathtt{C_2})\ \{\ \Pi(\bar{\mathtt{C}}_\mathtt{i})\ \bar{\mathtt{f}}_\mathtt{i};\ \Pi(\mathtt{K})\ \overline{\Pi(\mathtt{M})}\ \}$ in client code. And thus in updated client code, we have $\Pi(\mathtt{C_1}) <: \Pi(\mathtt{C_2})$.

Case-2: $\mathtt{C_1}$ is defined in $\mathtt{API_s}$.

In this case, $\mathtt{C_2}$ must also be defined in $\mathtt{API_s}$, and according to rule T-Π in SWIN type system. There exists $\mathtt{C_1} \hookrightarrow \mathtt{D_1}$, and $\mathtt{C_2} \hookrightarrow \mathtt{D_2}$ in $\mathtt{TypeMapping}(\Pi)$, and $\mathtt{D_1} <: \mathtt{D_2}$. Thus we have $\mathtt{D_1} = \pi(\mathtt{C_1}) <: \pi(\mathtt{C_2}) = \mathtt{D_2}$.

With both case proved, the lemma is proved. $\square$

2.4 Lemma 3

Suppose a FJ term $\mathtt{t}$ is well typed under environment $\Gamma = \Gamma_\mathtt{n}, \{\bar{\mathtt{x}} : \bar{\mathtt{C}}_\mathtt{x}\}$, i.e. $\Gamma \vdash \mathtt{t} : \mathtt{C_t}$, then after substituting variables $\bar{\mathtt{x}}$ with terms $\bar{\mathtt{t}}_\mathtt{v}$, with type $\Gamma_\mathtt{n} \vdash \bar{\mathtt{t}}_\mathtt{v} : \bar{\mathtt{C}}_\mathtt{v}$ and $\bar{\mathtt{C}}_\mathtt{v} <: \bar{\mathtt{C}}_\mathtt{x}$, $\mathtt{t}$ can be typed to $\mathtt{C_t}$ or a sub-class of $\mathtt{C_t}$. Namely,

$$\Gamma_\mathtt{n}, \{\bar{\mathtt{x}} : \bar{\mathtt{C}}_\mathtt{x}\} \vdash \mathtt{t} : \mathtt{C_t} \Longrightarrow \Gamma_\mathtt{n} \vdash [\bar{\mathtt{x}} \rightarrow \bar{\mathtt{t}}_\mathtt{u}]\mathtt{t} : \mathtt{C}'_\mathtt{t},\ \mathtt{C}'_\mathtt{t} <: \mathtt{C_t}$$

Proof: By induction on the derivation of a FJ term t. Then there are five cases to discuss.

Case-1 $\mathtt{t} = \mathtt{x}, \Gamma \vdash \mathtt{t} : \mathtt{C_t}, \mathtt{x} : \mathtt{C_t}$

In this case, we substitute $\mathtt{x}$ with a FJ term $\mathtt{t_u}$ of type $\mathtt{C_u}$ and $\mathtt{C_u} <: \mathtt{C_t}$. Then $\Gamma \vdash [\mathtt{x} \rightarrow \mathtt{t_u}]\mathtt{t} : \mathtt{C_u}$ and $\mathtt{C_u} <: \mathtt{C_t}$.

Case-2 $\mathtt{t} = (\mathtt{C})\mathtt{t_1},\ \Gamma \vdash \mathtt{t} : \mathtt{C}$.

This is done right away, as after substitution these still a cast to keep the term with type $\mathtt{C}$, thus $\Gamma \vdash [\bar{\mathtt{x}} \rightarrow \bar{\mathtt{t}}_\mathtt{u}]\mathtt{t} : \mathtt{C}$

Case-3 $\mathtt{t} = \mathtt{t_1.f},\ \Gamma \vdash \mathtt{t} : \mathtt{C_t},\ \mathtt{t_1} : \mathtt{C_1}$

By induction, we have $\Gamma \vdash \mathtt{t_1} : \mathtt{C}'_1$ and $\mathtt{C}'_1 <: \mathtt{C_1}$. Then the field access is also refered to the field in $\mathtt{C_1}$ (the supper class). And after substitution, we still have $\Gamma \vdash [\bar{\mathtt{x}} \rightarrow \bar{\mathtt{t}}_\mathtt{u}]\mathtt{t} : \mathtt{C_t}$.

Case-4 $\mathtt{t} = \mathtt{new}\ \mathtt{C_0}(\bar{\mathtt{t}}_\mathtt{x}),\ \Gamma \vdash \mathtt{t_x} : \mathtt{C_x}, \mathtt{t} : \mathtt{C_0}$

The substitution $\Gamma \vdash [\bar{\mathtt{x}} \rightarrow \bar{\mathtt{t}}_\mathtt{u}]\mathtt{t} = \mathtt{new}\ \mathtt{C_0}([\bar{\mathtt{x}} \rightarrow \bar{\mathtt{t}}_\mathtt{u}]\bar{\mathtt{t}}_\mathtt{x})$. By induction, after substitution, we have $\Gamma \vdash [\bar{\mathtt{x}} \rightarrow \bar{\mathtt{t}}_\mathtt{u}]\bar{\mathtt{t}}_\mathtt{x} : \bar{\mathtt{C}}'_\mathtt{x}$ and $\bar{\mathtt{C}}'_\mathtt{x} <: \bar{\mathtt{C}}_\mathtt{x}$. As the $\mathtt{t}$ can be well typed in Γ, by rule T-NEW (FJ-typing rule), We have

$$\frac{\texttt{fields}(\texttt{C}_0) = \bar{\texttt{D}}\ \bar{\texttt{f}} \qquad \Gamma \vdash \bar{\texttt{t}}_\texttt{x} : \bar{\texttt{C}}_\texttt{x} \qquad \bar{\texttt{C}}_\texttt{x} <: \bar{\texttt{D}}}{\Gamma \vdash \texttt{new C}(\bar{\texttt{t}}_\texttt{x}) : \texttt{C}_0}$$

And after substitution, we have

$$\frac{\texttt{fields}(\texttt{C}_0) = \bar{\texttt{D}}\ \bar{\texttt{f}} \qquad \Gamma \vdash [\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\bar{\texttt{t}}_\texttt{x} : \bar{\texttt{C}}'_\texttt{x} \qquad \bar{\texttt{C}}'_\texttt{x} <: \bar{\texttt{C}}_\texttt{x} <: \bar{\texttt{D}}}{\Gamma \vdash \texttt{new C}([\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\bar{\texttt{t}}_\texttt{x}) : \texttt{C}_0}$$

Then $\texttt{t} = \texttt{new C}([\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\bar{\texttt{t}}_\texttt{x})$ can also be typed to $\texttt{C}_0$.

Case-5 $\texttt{t} = \texttt{t}_0.\texttt{m}(\bar{\texttt{t}}_\texttt{x}),\ \Gamma \vdash \texttt{t}_\texttt{x} : \texttt{C}_\texttt{x}, \texttt{t}_0 : \texttt{C}_0, \texttt{t} : \texttt{C}$
In this case, we have $[\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\texttt{t} = [\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\texttt{t}_0.\texttt{m}([\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\bar{\texttt{t}}_\texttt{x})$
By induction, after substitution, we have:

$$\Gamma \vdash [\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\bar{\texttt{t}}_\texttt{x} : \bar{\texttt{C}}'_\texttt{x}, [\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\texttt{t}_0 : \texttt{C}'_0$$

and $\bar{\texttt{C}}'_\texttt{u} <: \bar{\texttt{C}}_\texttt{u}, \texttt{C}'_0 <: \texttt{C}_0$. As the term $\texttt{t}$ can be typed to $\texttt{C}_0$ before substitution, there exists the following type inference rule:

$$\frac{\Gamma \vdash \texttt{t}_0 : \texttt{C}_0 \qquad \texttt{mtype}(\texttt{m}, \texttt{C}_0) = \bar{\texttt{D}} \to \texttt{C} \qquad \Gamma \vdash \bar{\texttt{t}}_\texttt{x} : \bar{\texttt{C}}_\texttt{x} \qquad \bar{\texttt{C}}_\texttt{x} <: \bar{\texttt{D}}}{\Gamma \vdash \texttt{t}_0.\texttt{m}(\bar{\texttt{t}}_\texttt{x}) : \texttt{C}}$$

As $\texttt{C}'_0 <: \texttt{C}_0$, we have $\texttt{mtype}(\texttt{m}, \texttt{C}'_0) = \texttt{mtype}(\texttt{m}, \texttt{C}_0)$. Then we have:

$$\frac{\Gamma \vdash [\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\texttt{t}_0 : \texttt{C}'_0 \qquad \texttt{mtype}(\texttt{m}, \texttt{C}'_0) = \bar{\texttt{D}} \to \texttt{C} \qquad \Gamma \vdash [\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\texttt{t}_0\bar{\texttt{t}}'_\texttt{x} : \bar{\texttt{C}}'_\texttt{x} \qquad \bar{\texttt{C}}'_\texttt{x} <: \bar{\texttt{C}}_\texttt{x} <: \bar{\texttt{D}}}{\Gamma \vdash [\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\texttt{t}_0.\texttt{m}([\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\bar{\texttt{t}}_\texttt{x}) : \texttt{C}}$$

Thus we have $\Gamma \vdash [\bar{\texttt{x}} \to \bar{\texttt{t}}_\texttt{u}]\texttt{t} : \texttt{C}$. And the property holds.

With all these five cases dicussed, we finish the proof. □

2.5 Lemma 4

Suppose we have well typed SWIN code Π, if a term in original type environment can be typed to $\texttt{C}$, then update adaption, the term is well typed and the type is a subtype of $\Pi(\texttt{C})$. i.e.

$$\Gamma_\texttt{o} \vdash \texttt{t} : \texttt{C} \Longrightarrow \Gamma_\texttt{n} \vdash \Pi(\texttt{t}) : \texttt{C}',\ \texttt{where } \texttt{C}' <: \Pi(\texttt{C})$$

Assumption We assume that we cannot access the field of an API class in client code without using a public method defined in API.

Proof: By induction on a derivation of a term $\texttt{t}$. At each step of the induction, we assume that the desired property holds for all sub-derivations. We give our proof based on the last step of the derivation, which can only be one of variables, field access, method invocation, object creation or cast.

Case-1 : $\mathtt{t} = \mathtt{x},\ \Gamma_\mathtt{o} \vdash \mathtt{t} : \mathtt{C_t}$

The derivation of a term $\mathtt{t}$ is only one step. Then $\mathtt{t}$ must be a variable bounded in the definition of the method body. Suppose the type of x is $\Gamma_\mathtt{o} \vdash \mathtt{x} : \mathtt{C_t}$, then according to Lemma 1, we have $\Gamma_\mathtt{n} \vdash \mathtt{x} : \Pi(\mathtt{C_t})$, which hold the property.

Case-2 : $\mathtt{t} = \mathtt{t_1.f},\ \Gamma_\mathtt{o} \vdash \mathtt{t} : \mathtt{C_t},\ \mathtt{t_1} : \mathtt{C_1}$.

According to the definition, the type of $\mathtt{f}$ is $\mathtt{C_t}$, i.e. $\Gamma_\mathtt{o} \vdash \mathtt{f} : \mathtt{C_t}$

In this case, according to the rule E-T-FILED, $\Pi(\mathtt{t}) = \Pi(\mathtt{t_1}).\mathtt{f}$. According to the assumption, the term $\mathtt{t_1}$ can only be a client-defined class. Thus we have $\Pi(\mathtt{C_1}) = \mathtt{C_1}$.

By induction, we have $\Gamma_\mathtt{n} \vdash \Pi(\mathtt{t_1}) : \mathtt{C}'_1,\ \mathtt{where}\ \mathtt{C}'_1 <: \Pi(\mathtt{C_1})$. And we have $\mathtt{C}'_1 <: \mathtt{C_1}$. And $\mathtt{t_1.f}$ is refered to the field access in super class $\mathtt{C_1}$.

The class definition of $\mathtt{C_1}$ is $\mathtt{class}\ \mathtt{C_{11}}\ \mathtt{extends}\ \mathtt{C_{12}}\ \{\bar{\mathtt{C}}_\mathtt{1i}\ \bar{\mathtt{f}}_\mathtt{1i};\ \mathtt{K}\ \bar{\mathtt{M}}\ \}$, according to the rule E-DECLARATION, we have the definition of the field $\mathtt{f}$ as $\Gamma_\mathtt{n} \vdash \Pi(\mathtt{C_t}) : \mathtt{f}$.

Then we have $\Gamma \vdash \Pi(\mathtt{t_1}).\mathtt{f} : \Pi(\mathtt{C_t})$. Thus term $\mathtt{t}$ preserves the property.

Case-3 : $\mathtt{t} = (\mathtt{C})\mathtt{t_1},\ \Gamma_\mathtt{o} \vdash \mathtt{t} : \mathtt{C}$.

By the rule E-TERM-CAST, we have $\Pi(\mathtt{t}) = (\Pi(\mathtt{C}))\ \Pi(\mathtt{t_1})$, and then the type of the term is $\Gamma_\mathtt{n} \vdash \Pi(\mathtt{t}) : \Pi(\mathtt{C})$.

Case-4 : $\mathtt{t} = \mathtt{new}\ \mathtt{C_0}(\bar{\mathtt{t}}_\mathtt{u}),\ \Gamma_\mathtt{o} \vdash \mathtt{t_u} : \mathtt{C_u}, \mathtt{t} : \mathtt{C_0}$

In this case, we have two sub-cases:

sub-case 1 : The constructor $\mathtt{C_0}\ (\bar{\mathtt{C}}_2\ \bar{\mathtt{x}})$ is defined in client code rather than API, and class $\mathtt{C_0}$ has constructor $\mathtt{C_0}\ (\bar{\mathtt{C}}_2\ \bar{\mathtt{x}})$.

In this subcase, there will be no rule π in Π matching this term as a rule only matches an API usage. Then according to the rule E-ALTERNATIVE-NEW, $\Pi(\mathtt{t}) = \mathtt{new}\ \mathtt{C_0}(\overline{\Pi(\mathtt{t_u})})$. Now we need to prove that $\Pi(\mathtt{t})$ is well typed in $\Gamma_\mathtt{n}$ and this can be done with the following properties:

1. According to the rule E-CONSTRUCTOR, we have the constructor of class $\mathtt{C_0}$ after update is $\mathtt{C_0}(\overline{\Pi(\mathtt{C_2})}\ \mathtt{x})$, and $\mathtt{fields}(\mathtt{C_0}) = \overline{\Pi(\mathtt{C_2})}$.

2. As term $\mathtt{t}$ is well typed in the original code. We have the relationship $\bar{\mathtt{C}}_\mathtt{u} <: \bar{\mathtt{C}}_2$.

3. By induction, $\Gamma_\mathtt{n} \vdash \overline{\Pi(\mathtt{t_u})} : \bar{\mathtt{C}}'_\mathtt{u},\ \mathtt{where}\ \bar{\mathtt{C}}'_\mathtt{u} <: \overline{\Pi(\mathtt{C_u})}$.

4. As Π is well typed in SWIN type system, according to Lemma 2, $\overline{\Pi(\mathtt{C_u})} <: \overline{\Pi(\mathtt{C_2})}$.

With these four properties, we can have the term $\Pi(\mathtt{t})$ well typed according to typing rule T-NEW of FJ:

$$\frac{\mathtt{fields}(\mathtt{C_0}) = \overline{\Pi(\mathtt{C_2})} \qquad \Gamma_n \vdash \overline{\Pi(\mathtt{t_u})} : \bar{\mathtt{C}}'_u \qquad \bar{\mathtt{C}}'_u <: \overline{\Pi(\mathtt{C_u})} <: \overline{\Pi(\mathtt{C_2})}}{\Gamma_n \vdash \mathtt{new}\ \mathtt{C_0}(\overline{\Pi(\mathtt{t_u})}) : \mathtt{C_0}}$$

And then $\Pi(\mathtt{t})$ is well typed and has type $\mathtt{C_0}$, which is of course a subtype of $\mathtt{C_0}$.

sub-case 2 : $\mathtt{C_0}\ (\bar{\mathtt{C}}_2\ \bar{\mathtt{x}})$ is a constructor defined in $\mathtt{API_s}$, and the constructor has type $\bar{\mathtt{C}}_2 \to \mathtt{C_0}$.

As Π is well typed in SWIN type system, we must have $\mathtt{C_0} \hookrightarrow \mathtt{D_0} \in \mathtt{TypeMapping}(\Pi)$ (By T-Π), then there must be a rule $\pi = [(\overline{\mathtt{x} : \mathtt{C_x} \hookrightarrow \mathtt{D_x}})\ \mathtt{new}\ \mathtt{C_0}(\bar{\mathtt{x}}) : \mathtt{C_0} \to \mathtt{t_r} : \mathtt{D_0}]$ matching this term. Now we need to prove that after meta-variable substitution (We refer $\mathtt{t_r}$ after meta-variable substitution as $\sigma\mathtt{t_r}$), $\sigma\mathtt{t_r}$ is well typed under Γ_n, and $\Gamma_n \vdash \sigma\mathtt{t_r} : \mathtt{C_{tr}},\ \mathtt{where}\ \mathtt{C_{tr}} <: \mathtt{D_0}$. And this can be proved with the following properties:

1. According to T-R, $\{\mathtt{API_d}, \bar{\mathtt{x}} : \bar{\mathtt{D}}_x\} \vdash_{\mathtt{FJ}} \mathtt{t} : \mathtt{C_d}$.
2. According to E-NEW, $\bar{\mathtt{C}}_u <: \bar{\mathtt{C}}_x$
3. According to Lemma 2 and property 2, $\overline{\Pi(\mathtt{C_u})} <: \overline{\Pi(\mathtt{C_x})} = \bar{\mathtt{D}}_x$
4. By induction, $\Gamma_n \vdash \overline{\Pi(\mathtt{t_u})} : \bar{\mathtt{C}}'_u,\ \mathtt{where}\ \bar{\mathtt{C}}'_u <: \overline{\Pi(\mathtt{C_u})}$.

With these four properties, by E-T-NEW, after the application of substitution $\{\bar{\mathtt{x}} \to \overline{\Pi(\mathtt{t_u})}\}$ and $\overline{\Pi(\mathtt{t_u})} : \bar{\mathtt{C}}'_u,\ \mathtt{where}\ \bar{\mathtt{C}}'_u <: \overline{\Pi(\mathtt{C_u})} <: \bar{\mathtt{D}}_x$, then according to Lemma 3, after substitution, the type of return term should be $\mathtt{D}'_0$ and $\mathtt{D}'_0 <: \mathtt{D_0}$. Thus we have $\Gamma_n \Pi(\mathtt{t}) : \mathtt{D}'_0,\ \mathtt{where}\ \mathtt{D}'_0 <: \mathtt{D_0} = \Pi(\mathtt{C_0})$.

With these two sub cases proved, Case-4 is proved.

Case-5 $\mathtt{t} = \mathtt{t_0.m}(\bar{\mathtt{t}}_u),\ \Gamma_o \vdash \mathtt{t_u} : \mathtt{C_u}, \mathtt{t_0} : \mathtt{C_0}, \mathtt{t} : \mathtt{C}$

This case can also be divided into two subcases, which are similar to those of Case-4.

sub-case 1 : The method is defined in a client-defined class, i.e. $\mathtt{C_0}$ is a client-defined class. Then the application of Π will be evaluated with rule E-ALTER-INVOKE, and $\Pi(\mathtt{t}) = \Pi(\mathtt{t_0}).\mathtt{m}(\overline{\Pi(\mathtt{t_u})})$. As the term is well typed in original client code, we have:

$$\frac{\Gamma_o \vdash \mathtt{t_0} : \mathtt{C_0} \qquad \mathtt{mtype}(\mathtt{m}, \mathtt{C_0}) = \bar{\mathtt{D}} \to \mathtt{C} \qquad \Gamma_o \vdash \bar{\mathtt{t}}_u : \bar{\mathtt{C}}_u \qquad \bar{\mathtt{C}}_u <: \bar{\mathtt{D}}}{\Gamma_o \vdash \mathtt{t_0.m}(\bar{\mathtt{t}}_u) : \mathtt{C}}$$

Now we have the following properties:

1. By induction, $\Gamma_n \vdash \Pi(\mathtt{t_0}) : \mathtt{C}'_0,\ \mathtt{where}\ \mathtt{C}'_0 <: \Pi(\mathtt{C_0}) = \mathtt{C_0}$ (As $\mathtt{C_0}$ is defined in client code, we have $\Pi(\mathtt{C_0}) = \mathtt{C_0}$).

2. By induction, $\Gamma_n \vdash \overline{\Pi(t_u)} : \bar{C}'_u$, where $\bar{C}'_u <: \overline{\Pi(C_u)}$

3. According to Lemma 2, we have $\overline{\Pi(C_u)} <: \overline{\Pi(D)}$

4. As $\Pi(t_0) : C'_0$ is a subtype of C_0, we have $\mathtt{mtype}(m, C'_0) = \overline{\Pi(D)} \to \Pi(C)$ by E-METHOD.

Then with these four properties, we have:

$$\frac{\Gamma_n \vdash \Pi(t_0) : C'_0 \qquad \mathtt{mtype}(m, C'_0) = \overline{\Pi(D)} \to \Pi(C) \qquad \Gamma_n \vdash \overline{\Pi(t_u)} : \bar{C}'_u \qquad \bar{C}'_u <: \overline{\Pi(C_u)} <: \overline{\Pi(D)}}{\Gamma_n \vdash \Pi(t_0).m(\overline{\Pi(t_u)}) : \Pi(C)}$$

And thus sub-case 1 is proved.

sub-case 2 : The function is defined in an API class, and by rule T-Π, we have a rule $\pi = [(\overline{x : C_x \hookrightarrow D_x}, y : C_0 \hookrightarrow D_0)\ y.m(\bar{x}) : C \to t_r : D]$ to transform the method invocation, and we suppose the method type is $\mathtt{mtype}(m, C_0, API_s) = \bar{D} \to C$. As we have:

$$\frac{\Gamma_o \vdash t_0 : C_0 \qquad \mathtt{mtype}(m, C_0) = \bar{D} \to C \qquad \Gamma_o \vdash \bar{t}_u : \bar{C}_u \qquad \bar{C}_u <: \bar{D}}{\Gamma_o \vdash t_0.m(\bar{t}_u) : C}$$

1. $\bar{C}_u <: \bar{C}_x$, as this rule matches the term.

2. By Lemma 2, we have $\overline{\Pi(C_u)} <: \overline{\Pi(C_x)} = \bar{D}_x$

3. According to T-R, $\{API_d, \bar{x} : \bar{D}_x\} \vdash_{FJ} t : C_d$.

4. By induction, we have $\Gamma_n \vdash \overline{\Pi(t_u)} : \bar{C}'_u$, where $\bar{C}'_u <: \overline{\Pi(C_u)}$

With these four properties, by E-T-INVOKE, after the application of substitution, we have $\{\bar{x} \to \overline{\Pi(t_u)}\}$ and $\overline{\Pi(t_u)} : \bar{C}'_u$, where $\bar{C}'_u <: \overline{\Pi(C_u)} <: \bar{D}_x$, then according to Lemma 3, after substitution, the type of return term should be D' and $D' <: D$, and this finishes the proof of sub-case 2.

With these two subcases proved, Case-5 is proved.

With all these five cases proved, the proof of the Lemma completes by induction. And thus $\Gamma_o \vdash t : C \Rightarrow \Gamma_n \vdash \Pi(t) : C'$, where $C' <: \Pi(C)$.

And we finish our proof about the safety property. □

2.6 Lemma 5

Method declaration and class declaration are well typed after code adaption. i.e.

$$\Pi(M) = \Pi(C_1)\ m(\Pi(\bar{C}_m)\ \bar{x})\ \{\mathtt{return}\ \Pi(t);\}$$

$$\Pi(CL) = \mathtt{class}\ \Pi(C_1)\ \mathtt{extends}\ \Pi(C_2)\ \{\ \Pi(\bar{C}_i)\ \bar{f}_i;\ \Pi(K)\ \overline{\Pi(M)}\ \}$$

are well typed with new API.

Proof : For $\mathtt{\Pi(M)}$, as it is well typed with old API ($\mathtt{API_s}$), we have

$$\frac{\mathtt{\bar{x} : \bar{C}, this : C \vdash t_0 : E_0} \qquad \mathtt{E_0 <: C_0} \qquad \mathtt{CT(C) = class\ C\ extends\ D\ \{...\}} \qquad \mathtt{override(m, D, \bar{C} \to C_0)}}{\mathtt{C_0\ m(\bar{C}\ \bar{x})\ \{return\ t_0;\}\ OK\ in\ C}}$$

Now we can prove the lemma with following properties:

1. According to Lemma 2, $\mathtt{\Pi(E_0) <: \Pi(C_0)}$.
2. According to Lemma 4, $\mathtt{\Gamma_n \vdash t_0 : E_0',\ where\ E_0' <: \Pi(E_0)}$
3. By E-DECLARATION and E-METHOD, the form of method definition co-variant with Π, thus $\mathtt{CT(\Pi(C)) = class\ \Pi(C)\ extends\ \Pi(D)\ \{...\}}$ and $\mathtt{override(m, \Pi(D), \overline{\Pi(C)})}$

Thus we have:

$$\frac{\mathtt{\bar{x} : \overline{\Pi(C)}, this : \Pi(C) \vdash \Pi(t_0) : E_0'} \qquad \mathtt{E_0' <: \Pi(E_0) <: \Pi(C_0)} \qquad \mathtt{CT(\Pi(C)) = class\ \Pi(C)\ extends\ \Pi(D)\ \{...\}} \qquad \mathtt{override(m, \Pi(D), \Pi(\bar{C}) \to \Pi(C_0))}}{\mathtt{\Pi(C_0)\ m(\overline{\Pi(C)}\ \bar{x})\ \{return\ \Pi(t_0);\}\ OK\ in\ \Pi(C)}}$$

And this proves the safety of method declaration.

For class declaration, as all method declaration are well typed, we just need to prove the consistency of the constructor for $\mathtt{\Pi(C)}$, and this is immediate by E-CONSTRUCTOR.

Then the lemma is proved and all method declaration and class declaration are well typed. □

2.7 Theorem

After adaption with well typed SWIN code, new client code $\mathtt{Code_n = \Pi(Code_o)}$ is well typed under FJ type system.

Proof : The adapted code is well typed can have two perspective:

1. All FJ terms after adaption are well typed. (By Lemma 4)
2. Any class declaration is well typed. (By Lemma 5)

And thus we have the theorem proved i.e. $\mathtt{Code_n}$ is well typed in FJ type system. □