Universidad Rey Juan Carlos

Defensa de Tesis Doctoral

Programa en Tecnologías de la Información y las Comunicaciones

Doctorando: Alejandro Valdezate Sánchez

Director: Dr. Rafael Capilla Sevilla

VARIABILIDAD DINÁMICA EN SISTEMAS SENSIBLES AL CONTEXTO

Contenido

Motivación

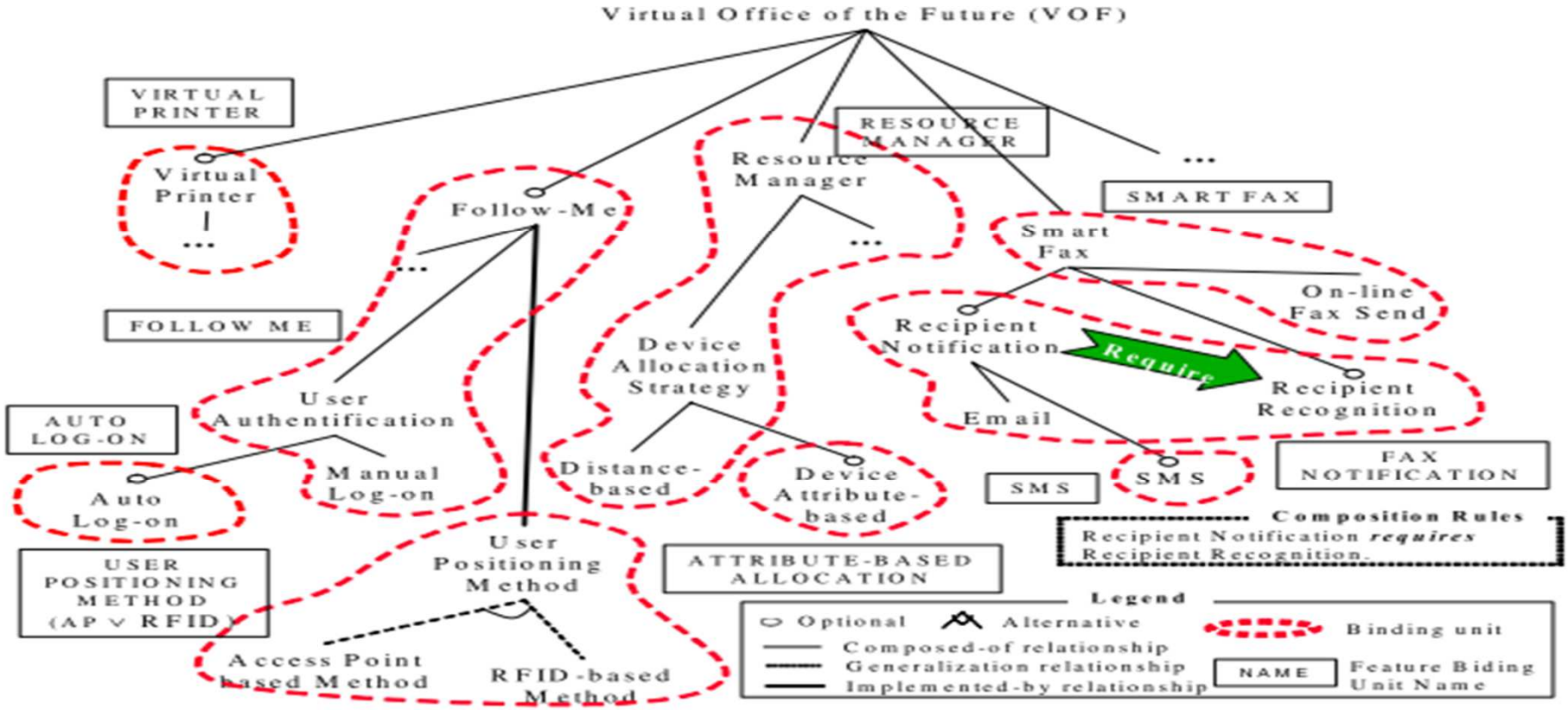
- La **variabilidad software** permite configurar características visibles de sistemas (denominados **variantes**) que cuentan con múltiples opciones de configuración.
- La **configuración de los variantes** puede realizarse en el lado del desarrollador de manera estática o en el lado del cliente de manera más dinámica.
- Los sistemas actuales dependientes del **contexto** requieren una gestión de la **variabilidad** que se pueda **automatizar** durante la ejecución del sistema.
- La gestión de la variabilidad en tiempo de ejecución se denomina **variabilidad dinámica**.

EXISTE UNA CARENCIA DE SOLUCIONES QUE PERMITAN GESTIONAR LA VARIABILIDAD DINÁMICA

Estado del arte

- Los **modelos de variabilidad software** surgidos en los ‘90 (FODA) se han enriquecido con semántica para permitir modelar los valores y atributos de las opciones configurables.
- Esta variabilidad permite configurar productos en el **espacio** (número y tipo de productos) y en el **tiempo**, a través del **binding time,** que es el momento en el cual los variantes toman sus valores en diferentes momentos.
- Los modelos de variabilidad necesitan modelar **restricciones** entre los diferentes variantes o características del producto para limitar el número de productos configurables.

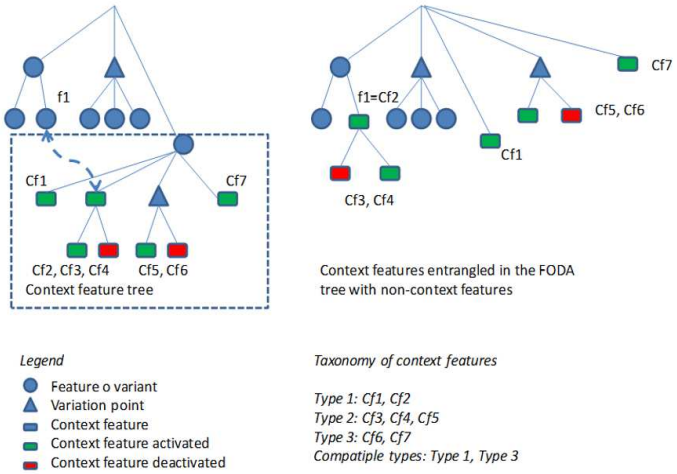
Estado del arte



LOS MODELOS DE VARIABILIDAD ESTÁTICOS SON INADECUADOS PARA SOPORTAR LOS ASPECTOS DINÁMICOS DE CIERTOS SISTEMAS

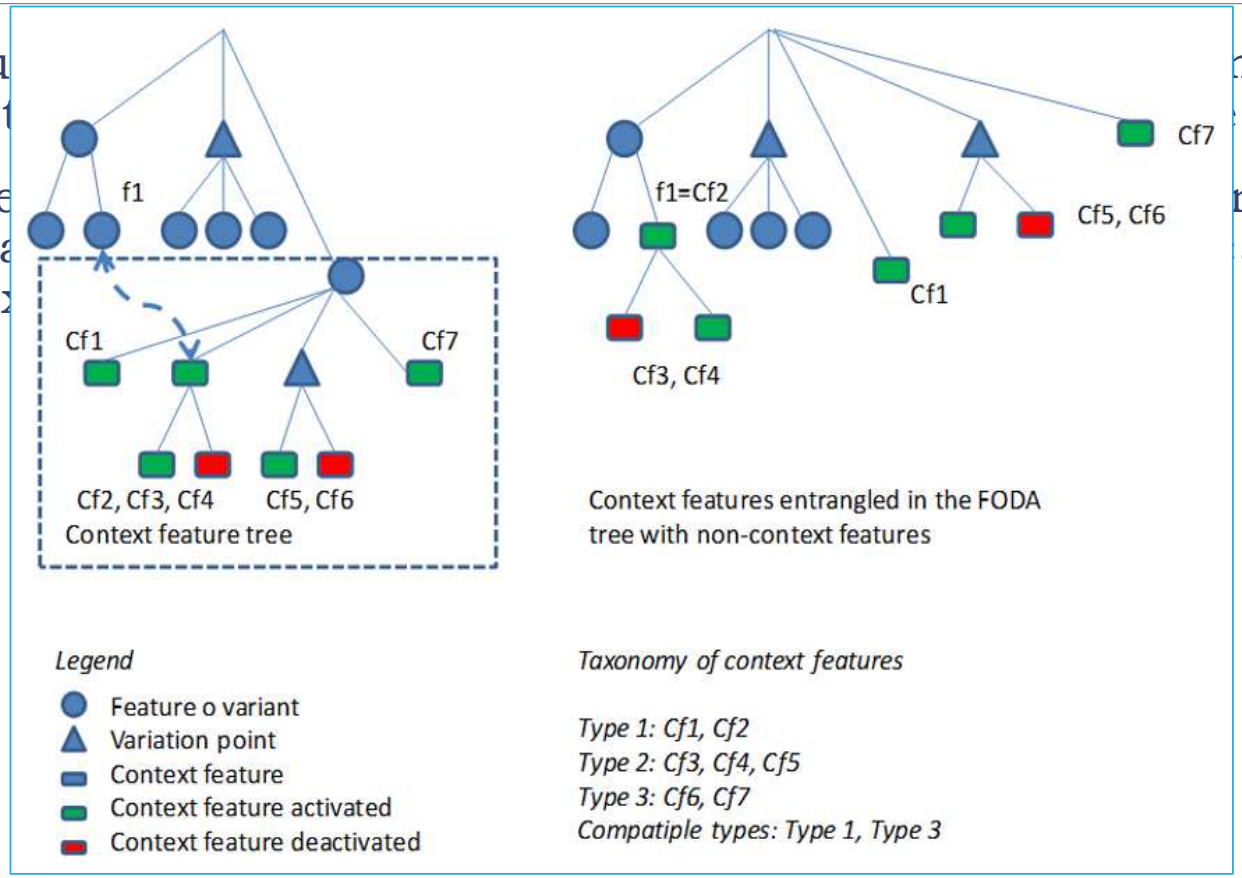
Estado del arte

- El uso de **variabilidad de contexto** permite modelar características en sistemas dependientes del contexto.
- Existen algunos lenguajes experimentales (Context-Oriented Programming – COP) que tratan de modelar cambios en el contexto en diferentes momentos.



Estado del arte

- El u… mite modelar caract… xto.
- Existe… ntext-Oriented Progra… ambios en el contex…

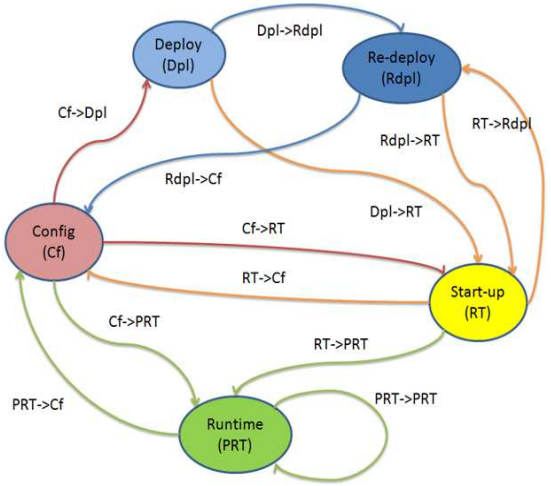


CADA VEZ ES MÁS FRECUENTE ENCONTRAR SISTEMAS QUE DEMANDAN VARIABILIDAD CONTEXTUAL

Estado del arte

- El **binding time dinámico** permite configurar variantes en tiempo de ejecución.
- Sistemas complejos pueden requerir transiciones entre modos de operación diferentes que den lugar a la necesidad de tener variantes con diferentes binding times.

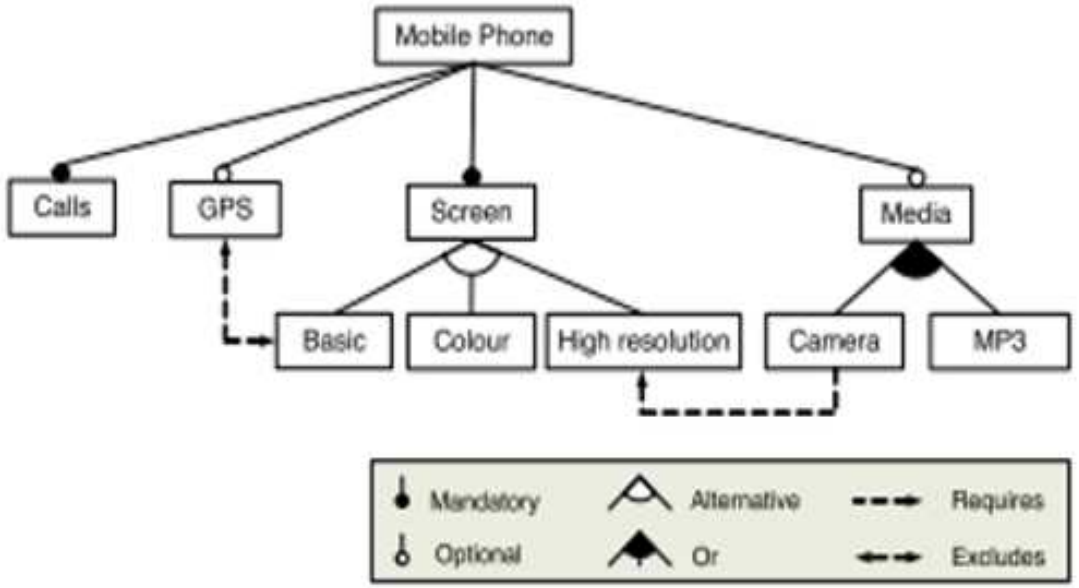
Binding time	Static/dynamic binding	Configurability	Single/multiple binding times	Binding on the developer/customer side	Transition for multiple binding times
Design	S	N/A	SI	D	N/A
Compilation/link	S	Low	SI	D	N/A
Build/assembly	S	Low	SI	D	N/A
Programming	S	Medium	SI	D	N/A
Configuration (Cf)	S/D	Medium/high	SI/MU	D/C	Cf→Dpl Cf→RT
Deploy (Dpl) and redeploy (Rdpl)	S/D	Medium/high	SI/MU	D/C	Dpl→Rdpl Rdpl→Cf
Runtime (RT) (start-up)	D	High	SI/MU	C	Dpl→RT Rdpl→RT RT→Rdpl RT→Cf
Pure runtime (PRT) (operational mode)	D	Very high	SI/MU	C	RT→PRT PRT→Cf Cf→PRT PRT→PRT



LAS LÍNEAS DE PRODUCTO SOFTWARE DINÁMICAS UTILIZAN TÉCNICAS DE VARIABILIDAD EN TIEMPO DE EJECUCIÓN Y BINDING DINÁMICO

Estado del arte

- Los sistemas de variabilidad tradicionales usan *solvers* para **procesar las restricciones** entre características de tipo «require» o «exclude».



AÑADIR Y ELIMINAR RESTRICCIONES ENTRE CARACTERÍSTICAS EN TIEMPO DE EJECUCIÓN SUPONE UN DESAFÍO PARA LOS SOLVERS ACTUALES

Estado del arte

- Las primeras experiencias de variabilidad dinámica corresponden a vehículos guiados autónomos donde la ruta dentro de una fábrica se puede modificar en tiempo de ejecución.
- Otras propuestas hacen referencia únicamente a la activación y desactivación de características con uso de sensores y sistemas ciber-físicos.
- Propuestas recientes (2022) aplican los conceptos de variabilidad dinámica para añadir variantes y valores en robots en tiempo de ejecución.

EXISTE UNA CARENCIA PARA GESTIONAR LOS CAMBIOS EN LA VARIABILIDAD ESTRUCTURAL EN TIEMPO DE POST-DESPLIEGUE

Planteamiento del problema

Subproblema 1
Modificar la variabilidad estructural en tiempo de ejecución

Subproblema 2
Clasificación de características en un árbol de variabilidad

Subproblema 3
Eliminación de características en un árbol de variabilidad

Subproblema 4
Comprobación de restricciones de forma automática en tiempo de ejecución

Subproblema 5
Automatización del tratamiento de puntos de variación

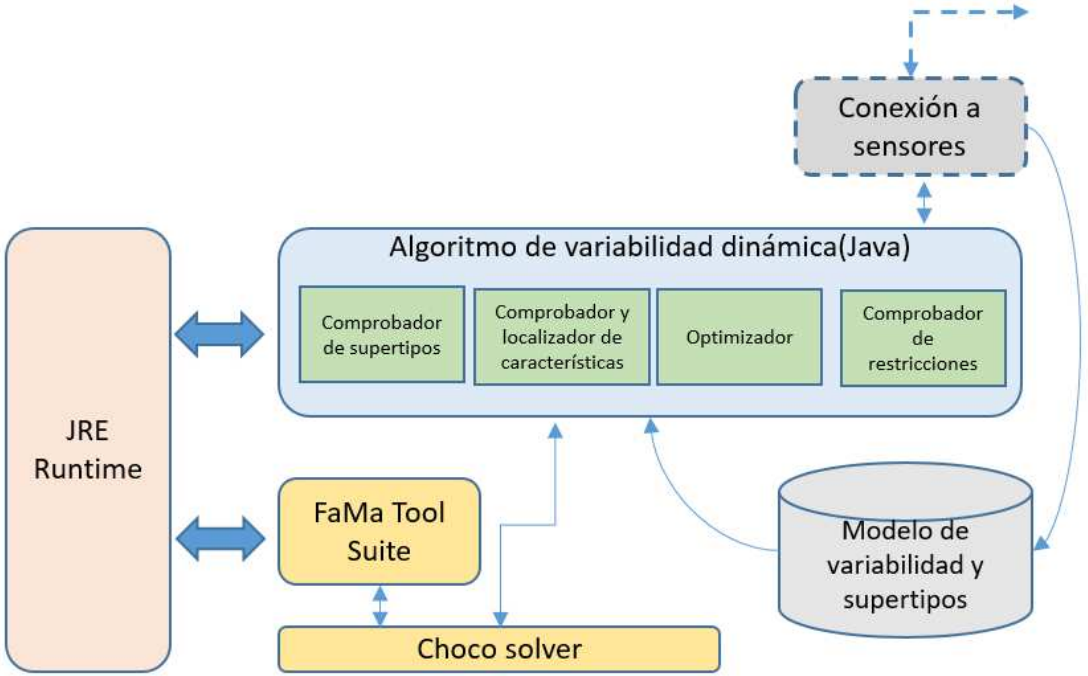
SE PRETENDE PROPORCIONAR UNA SOLUCIÓN DE VARIABILIDAD DINÁMICA QUE RESUELVA LA MODIFICACIÓN ESTRUCTURAL DE LA VARIABILIDAD EN TIEMPO DE EJECUCIÓN

Solución

Subproblema 1
Modificar la variabilidad estructural en tiempo de ejecución

Modelo conceptual para modificar la variabilidad en tiempo de ejecución

- Elementos que soportan el algoritmo de variabilidad dinámica.
- Integración con *solvers* de restricciones.
- Posibilidad de conexión a sistemas dependientes del contexto.



LA INTEGRACIÓN DE TODAS LAS PARTES PERMITE AUTOMATIZAR LA GESTIÓN DE CARACTERÍSTICAS

Solución

Subproblema 2
Clasificación de características en un árbol de variabilidad

Clasificación basada en supertipos

- Clasificación de características con la ayuda de **supertipos**.
- El uso de supertipos hace posible anticipar los lugares dónde es posible añadir una nueva característica y ayuda a decidir cuál es el más idóneo.
- Los supertipos actúan como clasificadores generales para los variantes, agrupándolos por funcionalidades.
- El diseñador del modelo de variabilidad debe definir sus propios supertipos.

$Comp\ (ST_A, ST_B) = 1$, si $ST_A = ST_B$ $\quad$ $Comp\ (ST_A, ST_B) = 0$, en otros casos

$ST(F_X) = \{ST_A, ST_B, ST_C \ldots ST_N\}$ $\quad$ $ST_{(ASISTENTE\_HIS)} = \{MULTIMEDIA, COMUNICACIONES\}$

DETERMINAR CUÁL ES LA UBICACIÓN ÓPTIMA DE NUEVAS CARACTERÍSTICAS PUEDE LOGRAR QUE LOS SISTEMAS FINALES ESTÉN MÁS EQUILIBRADOS

Solución

Subproblema 2
Clasificación de características en un árbol de variabilidad

Clasificación basada en supertipos

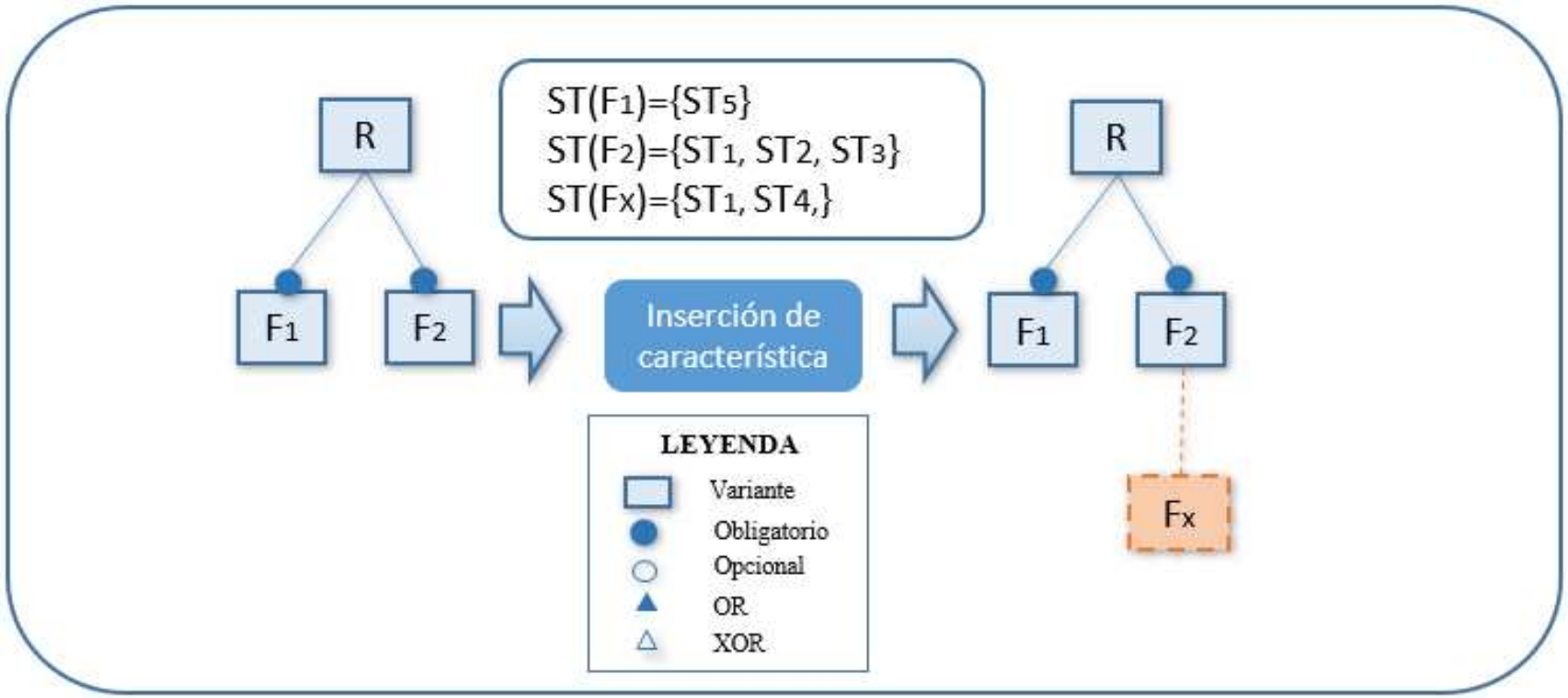
- Colocación de características en el lugar más adecuado gracias al **algoritmo de variabilidad dinámica**.
- Se contemplan cinco posibles escenarios para añadir características de forma dinámica.
- **Escenario 2**: Ocurre cuando insertamos una característica en un árbol de variabilidad donde el lugar de inserción no tiene hijos.
- **Escenario 3**: Ocurre cuando insertamos una característica en un árbol de variabilidad donde el lugar de inserción tiene hijos previamente.

Solución

Subproblema 2
Clasificación de características en un árbol de variabilidad

Clasificación basada en supertipos

- Colocación de características en el lugar más adecuado gracias a
- S
- I
- I



Solución

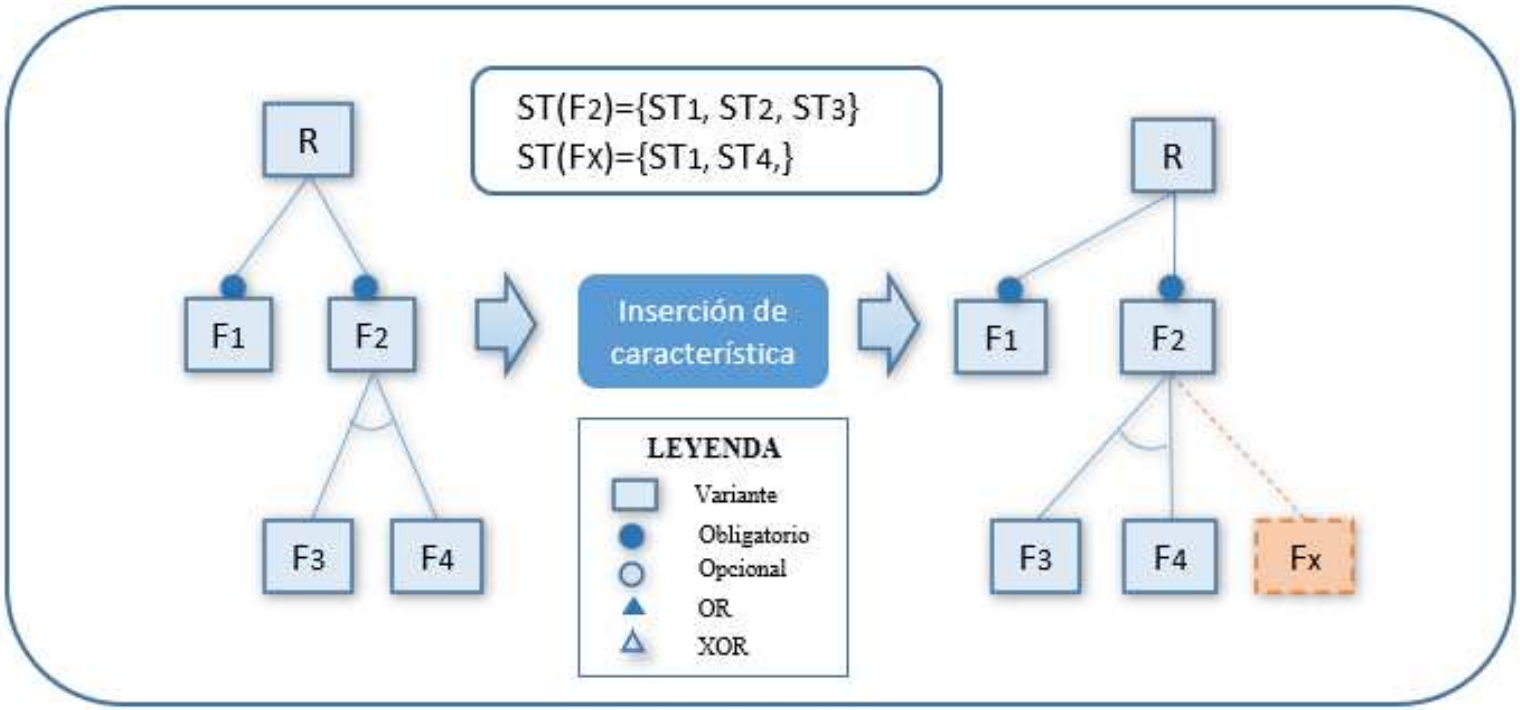
Subproblema 2
Clasificación de características en un árbol de variabilidad

Clasificación basada en supertipos

- Colocación de características en el lugar más adecuado gracias al **a**
- Se c
  car
- **Esc**
  un
  hijc
- **Esc**
  un
  pre

n

n
jos

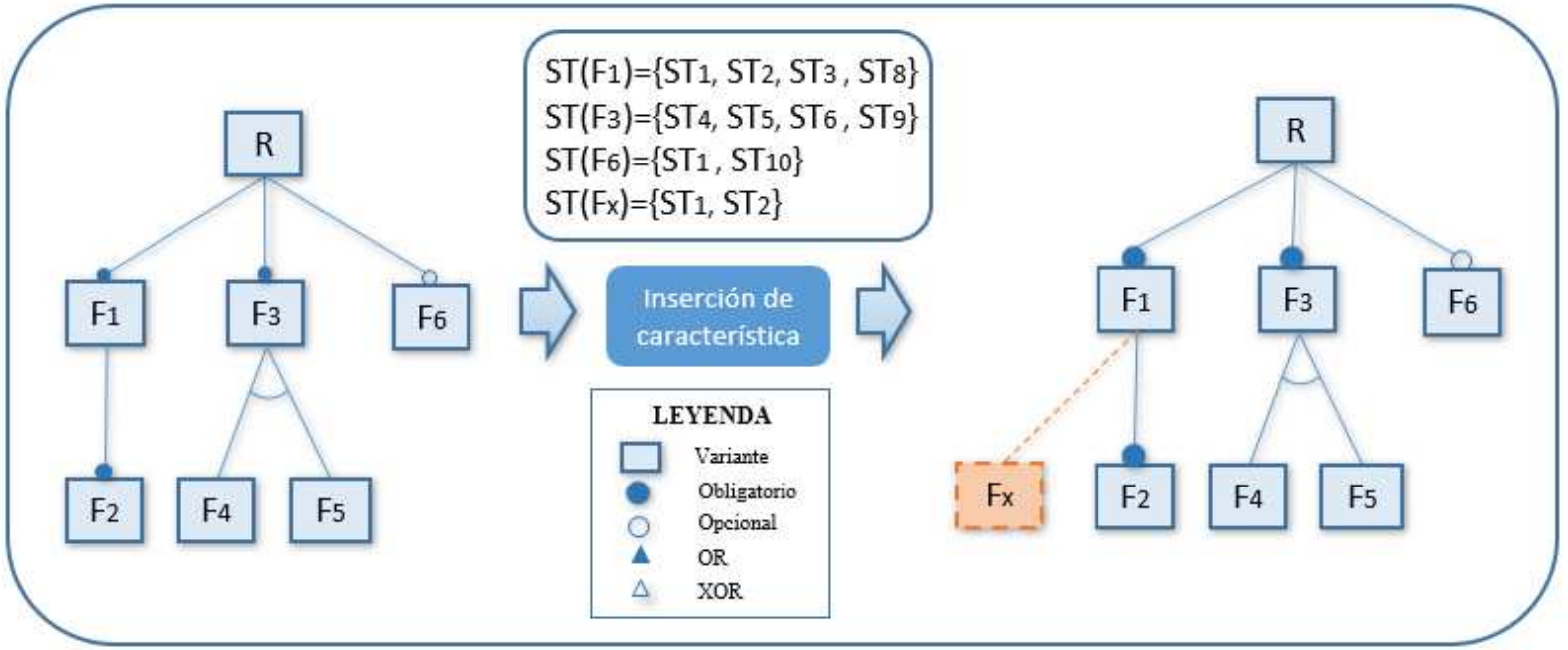


Solución

Subproblema 2
Clasificación de características en un árbol de variabilidad

Clasificación basada en supertipos

- Colocación de características en el lugar más adecuado gracias al **algoritmo de variabilidad dinámica**
- S
  c
- **I**
  y
  v
- **I**
  c
  r



rción

rción
la

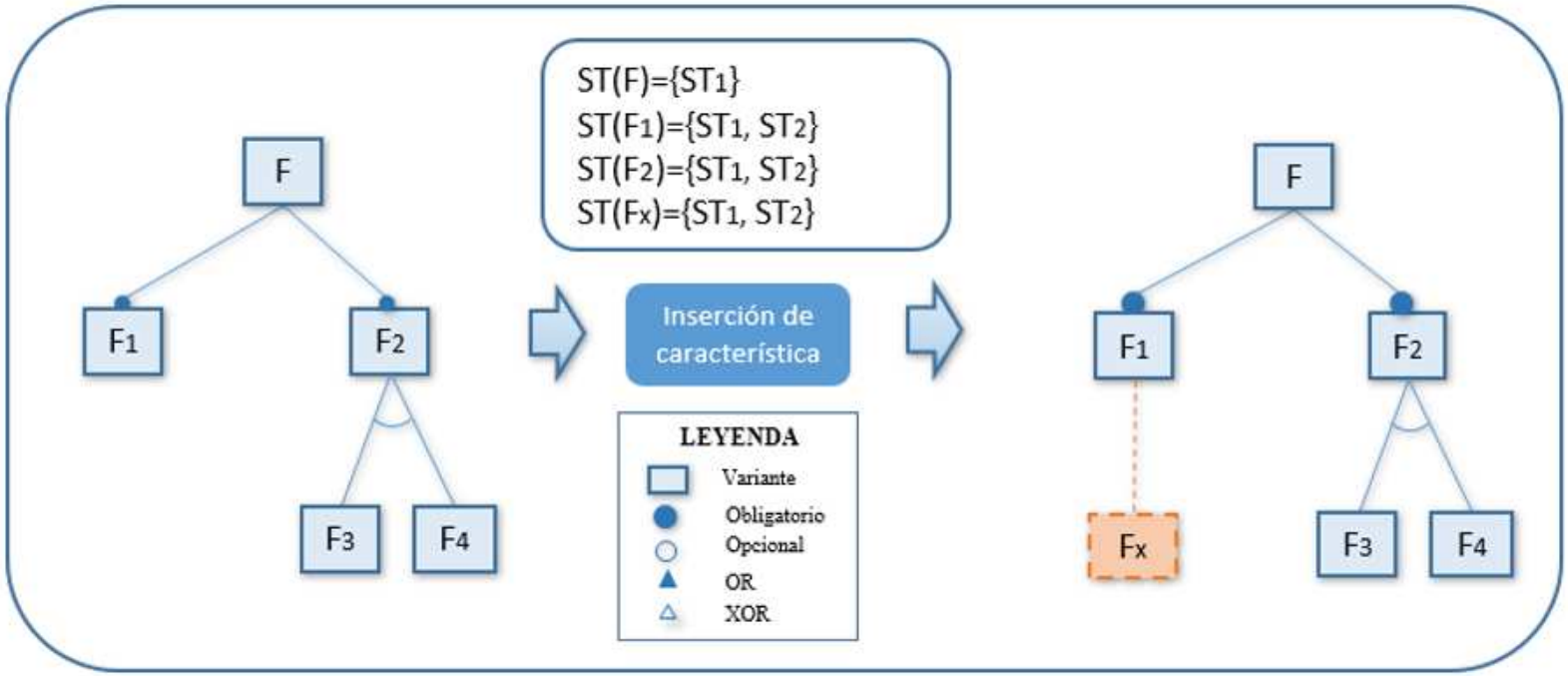
Universidad Rey Juan Carlos

Solución

Subproblema 2
Clasificación de características en un árbol de variabilidad

Clasificación basada en supertipos

- Colocación de características en el lugar más adecuado gracias al **algoritmo de variabilidad dinámica**.
- Se c
car
- **Esc**
y en
var
- **Esc**
con
ram



ón

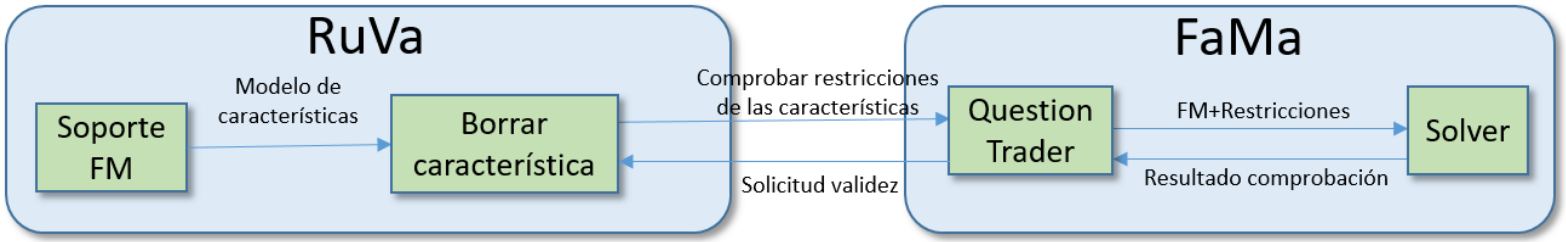
ón

Solución

Subproblema 3.
Eliminación de características en un árbol de variabilidad

Eliminación en cascada y restricciones

- No se precisa el modelo de supertipos para eliminar características.
- Sólo es necesario comprobar la lista de restricciones de la característica a eliminar.
- Los hijos de la característica a eliminar también desaparecen.
- La comprobación final de restricciones se realiza con el *solver* o bien detenemos el proceso si el modelo de variabilidad no es válido.
- El comprobador de restricciones distingue qué soluciones son válidas.
- Restricciones “require” y “exclude”.



Solución

Subproblema 5.
Automatización del tratamiento de puntos de variación.

Inserción dependiendo de la fórmula lógica de las características

- La **gestión de los puntos de variación** se hace en base al número de hijos del nodo candidato y sus relaciones.
- Las características con un solo hijo simplifican la elección de la relación resultante, siendo la nueva característica la que determine esta decisión.
- Se contemplan cuatro casos posibles.
- **Caso 2**: Ocurre cuando insertamos una característica con una relación de tipo AND en un árbol de variabilidad donde el lugar de inserción ya tiene una relación AND.
- **Caso 3**: Ocurre cuando insertamos una característica de tipo OR/XOR habiendo ya una relación OR/XOR.
- **Caso 4**: Sucede al insertar una característica obligatoria en un punto de inserción que ya tiene relaciones de tipo AND y OR/XOR.

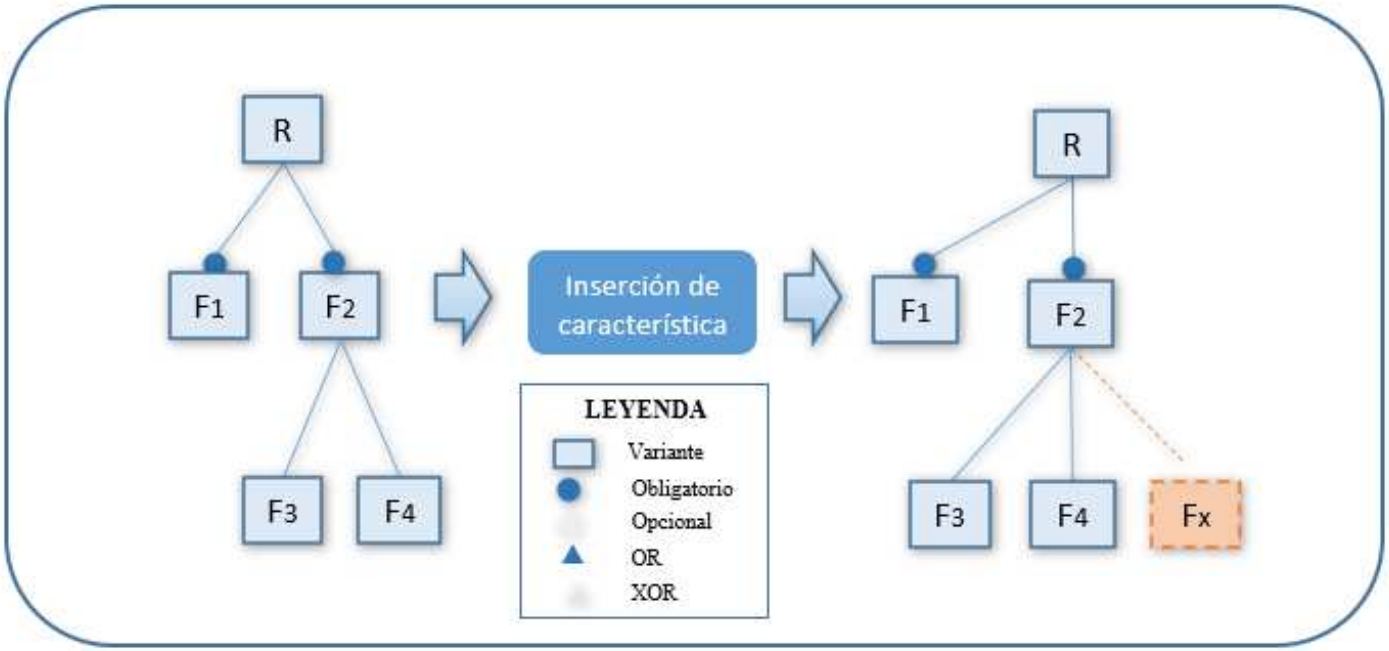
SI NO SE CONOCE LA RELACIÓN DE LA CARACTERÍSTICA A INSERTAR
¿CÓMO SE DECIDE DÓNDE INSERTARLA?

Solución

Subproblema 5.
Automatización del tratamiento de puntos de variación.

Localización basada en la relación

- La **gestión de los puntos de variación** se hace en base al número de hijos del nodo candidato y sus relaciones.
- Las car… de la relación… etermine esta de…
- Se cont…
- **Caso 2**: …na relación… ugar de inserci…
- **Caso 3**: …po OR/XOR habienc…
- **Caso 4**: …n un punto de inserción que ya tiene relaciones de tipo AND y OR/XOR.



SI NO SE CONOCE LA RELACIÓN DE LA CARACTERÍSTICA A INSERTAR
¿CÓMO SE DECIDE DÓNDE INSERTARLA?

Universidad Rey Juan Carlos

Solución

Subproblema 5.
Automatización del tratamiento de puntos de variación.

Localización basada en la relación

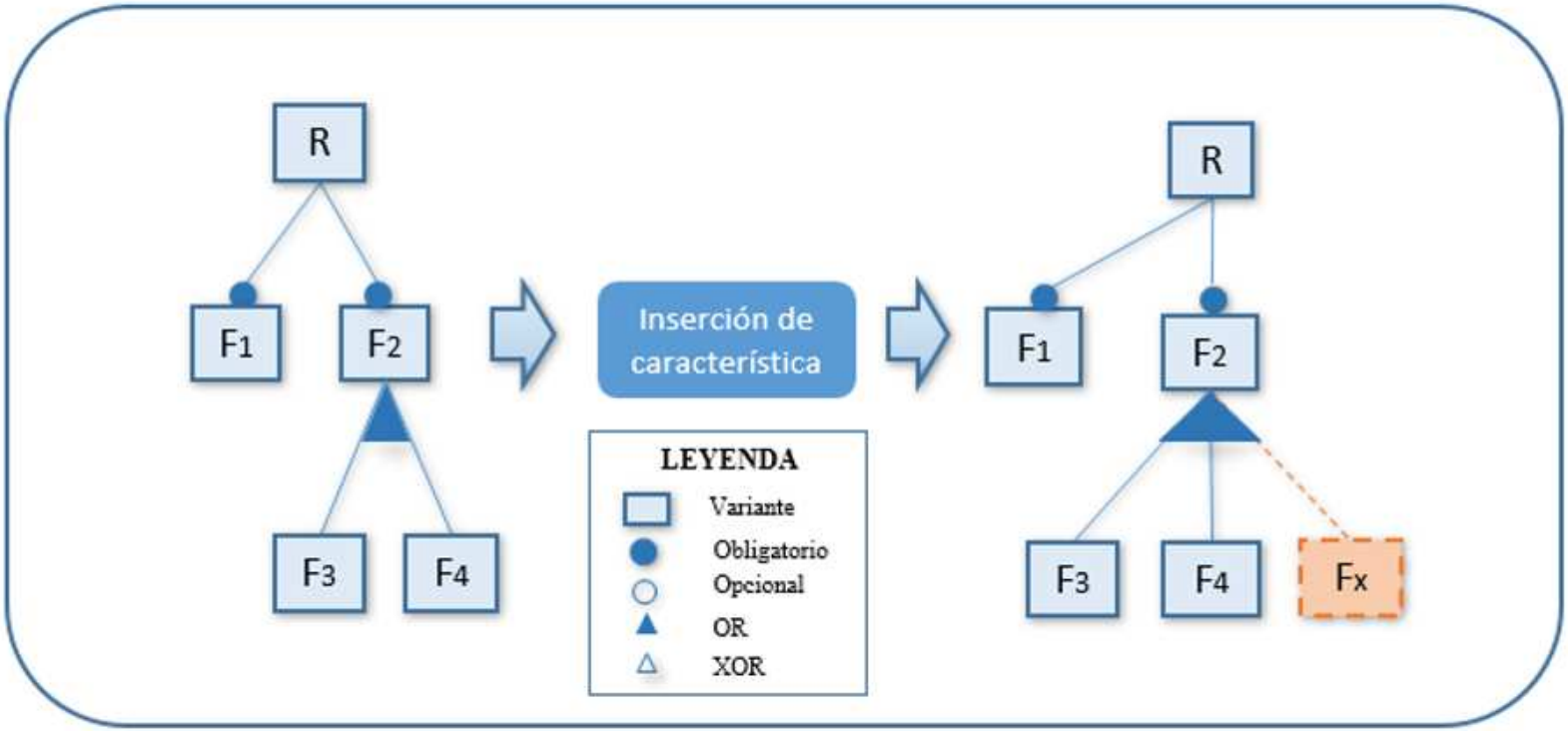
- La **gestión de los puntos de variación** se hace en base al número de hijos del nodo candidato y sus relaciones.
- Las
  rela
  esta
- Se
- **Cas**
  rela
  inse
- **Cas**
  hab
- **Cas**
  de i

la
rmine

ır de

OR/XOR

n punto



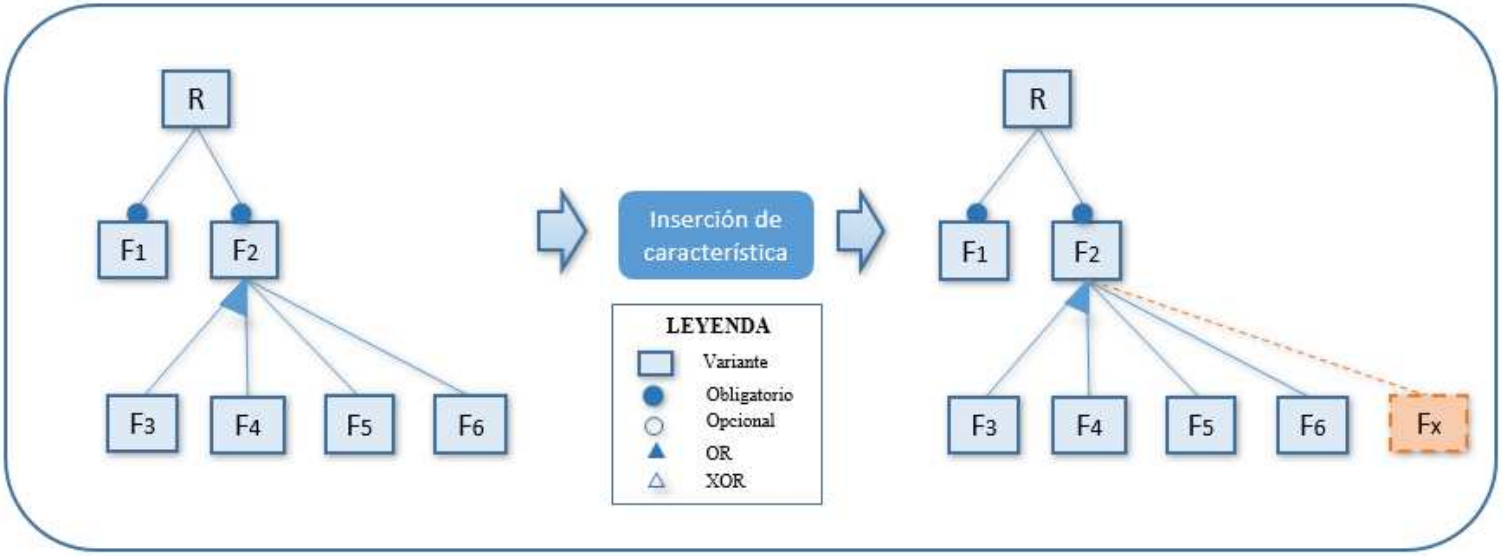
SI NO SE CONOCE LA RELACION DE LA CARACTERISTICA A INSERTAR
¿CÓMO SE DECIDE DÓNDE INSERTARLA?

Solución

Subproblema 5.
Automatización del tratamiento de puntos de variación.

Localización basada en la relación

- La **gestión de los puntos de variación** se hace en base al número de hijos del nodo candidato y sus relaciones.
- Las características con un solo hijo simplifican la elección de la relac… termine esta…
- Se cc…
- **Caso** … na relac… gar de inser…
- **Caso** … o OR/XOR habie…
- **Caso 4**: Sucede al insertar una característica obligatoria en un punto de inserción que ya tiene relaciones de tipo AND y OR/XOR.

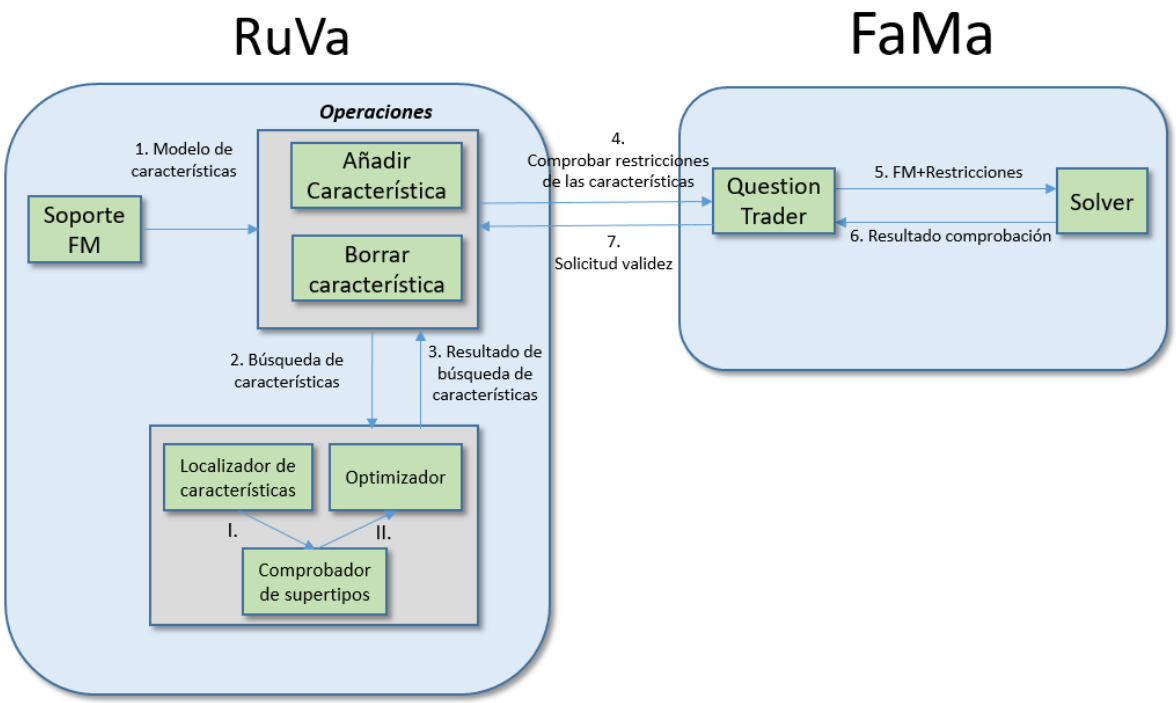


SI NO SE CONOCE LA RELACIÓN DE LA CARACTERÍSTICA A INSERTAR
¿CÓMO SE DECIDE DÓNDE INSERTARLA?

Solución

Arquitectura software para gestionar la variabilidad dinámica

- Operaciones soportadas por RuVa.
- Soporte de varios formatos de modelos de características e integración con diferentes sistemas.
- Resolución de restricciones en FaMa.



Experimentación

- Puesta a prueba en diferentes modelos de variabilidad con diferentes tamaños y composición (**Eficiencia**)
- Tests de rendimiento para comprobar los límites de aplicación del algoritmo (**Escalabilidad**).
- Simulación con un robot para evaluar el algoritmo en un sistema dependiente del contexto (**Eficiencia e Interoperabilidad**).

Experimentación

Parte 1
Validación de escenarios

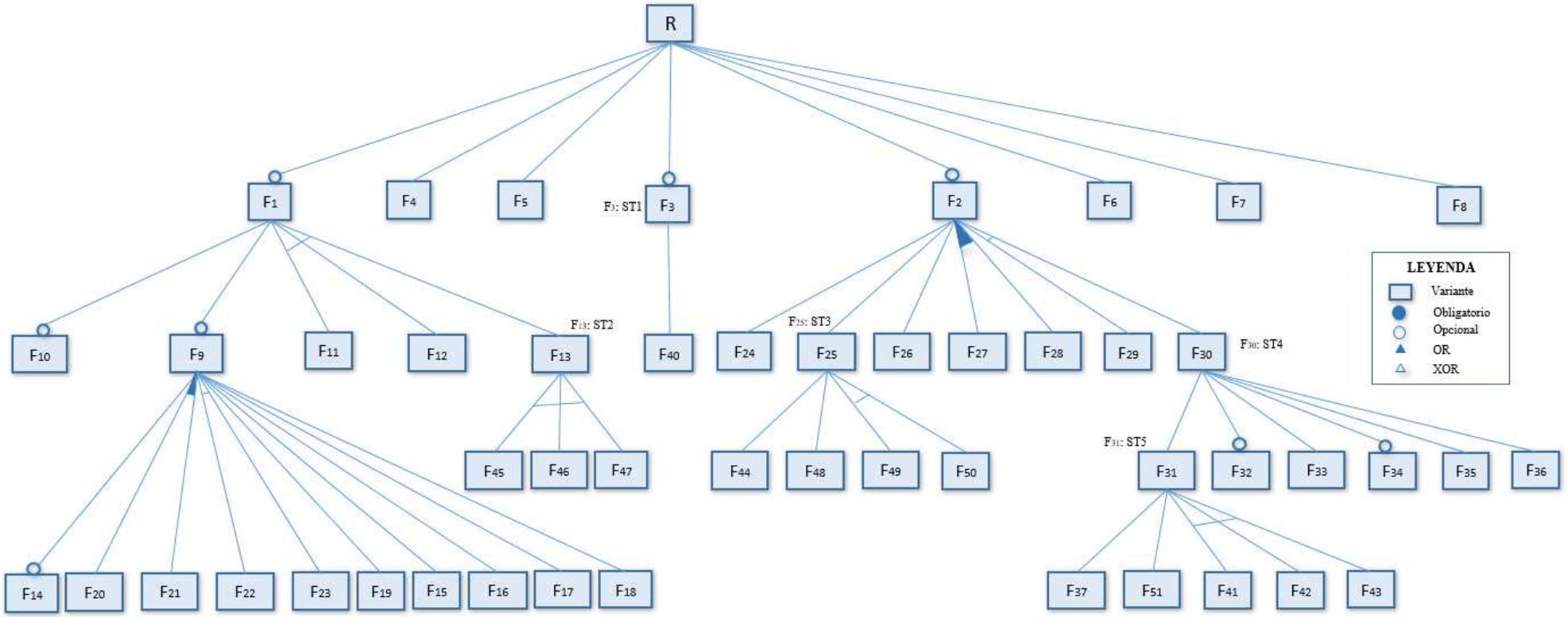
Pruebas de eficiencia

- Árbol de 50 nodos, con supertipos en 5 nodos. 1 o 2 supertipos compatibles por nodo, de 5 supertipos posibles.
- Al insertar 10 características nuevas Fx_n, estas se ubican en 4 de los nodos existentes.
- Gracias a la inserción guiada con supertipos, las 10 características quedan insertadas en el lugar más adecuado, manteniendo el equilibrio del árbol.

Experimentación

Parte 1
Validación de escenarios

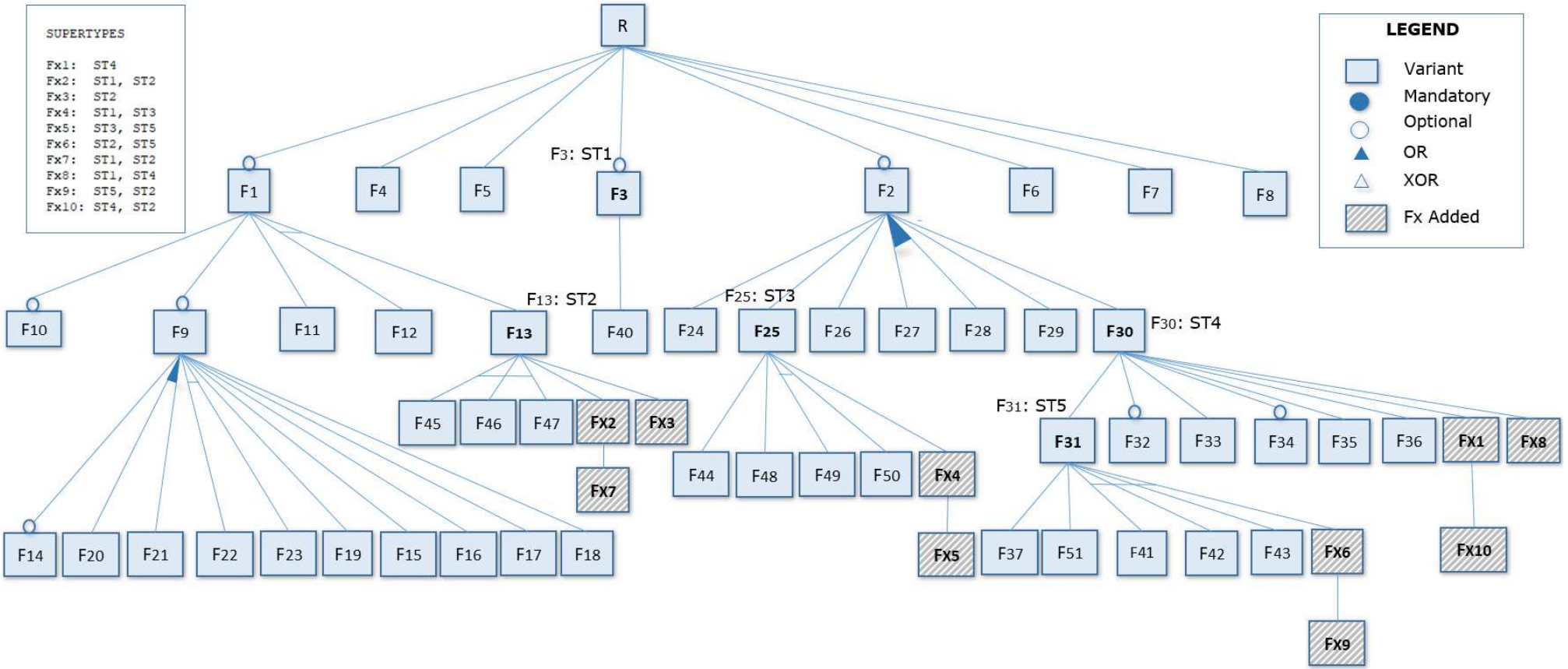
Pruebas de eficiencia



Experimentación

Parte 1
Validación de escenarios

Pruebas de eficiencia



LA INSERCION DE CARACTERÍSTICAS CON SUPERTIPOS GENÉRICOS ES SATISFACTORIA

Universidad Rey Juan Carlos

Experimentación

Parte 1
Validación de escenarios

Pruebas de eficiencia

- Los resultados dependen en gran medida de la estructura del modelo de variabilidad.

Juego de prueba	Tiempo (segundos)	Inserciones válidas	Inserciones no válidas	ratio
1-1	1,03	1	0	100,0%
1-2	0,99	1	0	100,0%
3-1	1,98	2	1	66,7%
3-2	0,96	1	2	33,3%
5-1	1,80	2	3	40,0%
5-2	2,65	3	2	60,0%
10-1	7,63	8	2	80,0%
10-2	6,89	8	2	80,0%

CON JUEGOS DE PRUEBA PEQUEÑOS NO ES POSIBLE EXTRAER CONCLUSIONES ADECUADAS

Experimentación

Parte 1
Validación de escenarios

Pruebas de eficiencia

- Repetición aleatoria de 100 juegos de prueba
- En **modelos** de variabilidad más **grandes** se logran mejores tasas de **éxito**.
- El tiempo medio de inserción no se ve muy afectado por los tamaños de los conjuntos de inserción.

Conjunto de prueba	Rangos de tiempo (s)	Tiempo medio (s)	Tasa media de éxito
Repeticiones de inserción de 1 característica	[0,688-1,183]	0,621	76%
Repeticiones de inserción de 3 características	[0,770-2,787]	0,678	90%
Repeticiones de inserción de 5 características	[1,363-3,631]	0,628	90%
Repeticiones de inserción de 10 características	[4,314-9,359]	0,741	92%

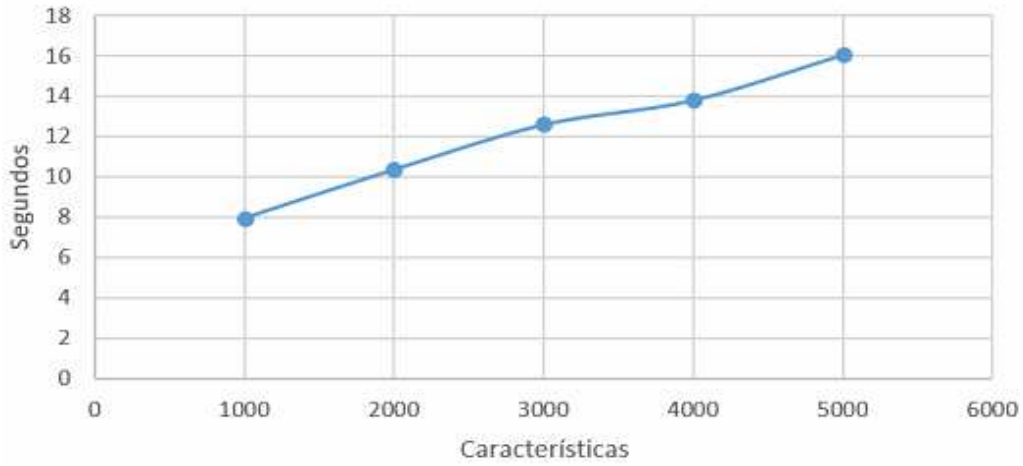
Experimentación

Parte 2
Escenarios de escalabilidad

Pruebas de rendimiento

- En modelos de variabilidad de gran tamaño como 5.000 características, el tiempo medio de respuesta se incrementa pero no es excesivo.

Tamaño FM	Tiempo (s)
1000	7,98
2000	10,38
3000	12,61
4000	13,82
5000	16,07



PARA SISTEMAS NO CRÍTICOS CON MODELOS DE VARIABILIDAD GRANDES EL ALGORITMO SE COMPORTA DE MANERA SATISFACTORIA

Universidad Rey Juan Carlos

Experimentación

Parte 3
Simulación con robot TurtleBot

Reconfiguración de la variabilidad en un robot

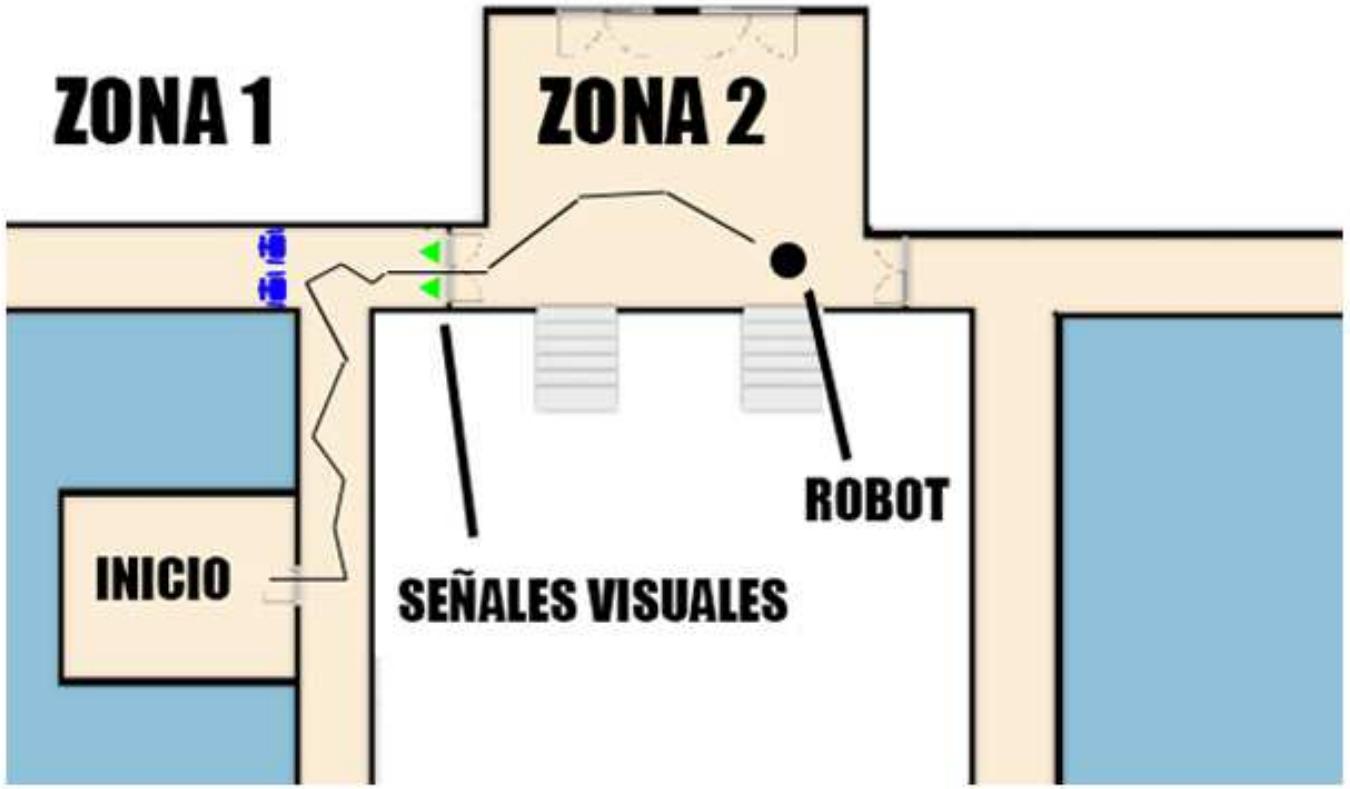
- Exploración de un área ubicada en la planta baja del edificio Departamental II mediante un robot.
- Comprende 2 zonas diferentes separadas.
- El robot realiza el recorrido siguiendo un algoritmo de exploración, pero al encontrar obstáculos cambia de ruta y llega a una nueva zona.

Experimentación

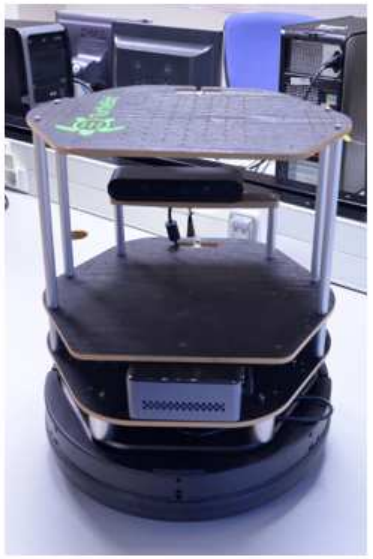
Parte 3
Simulación con robot TurtleBot

Reconfiguración de la variabilidad en un robot

- Explor… ficio Depart…
- Compr…
- El robo… xploración, pero al… a nueva zona.



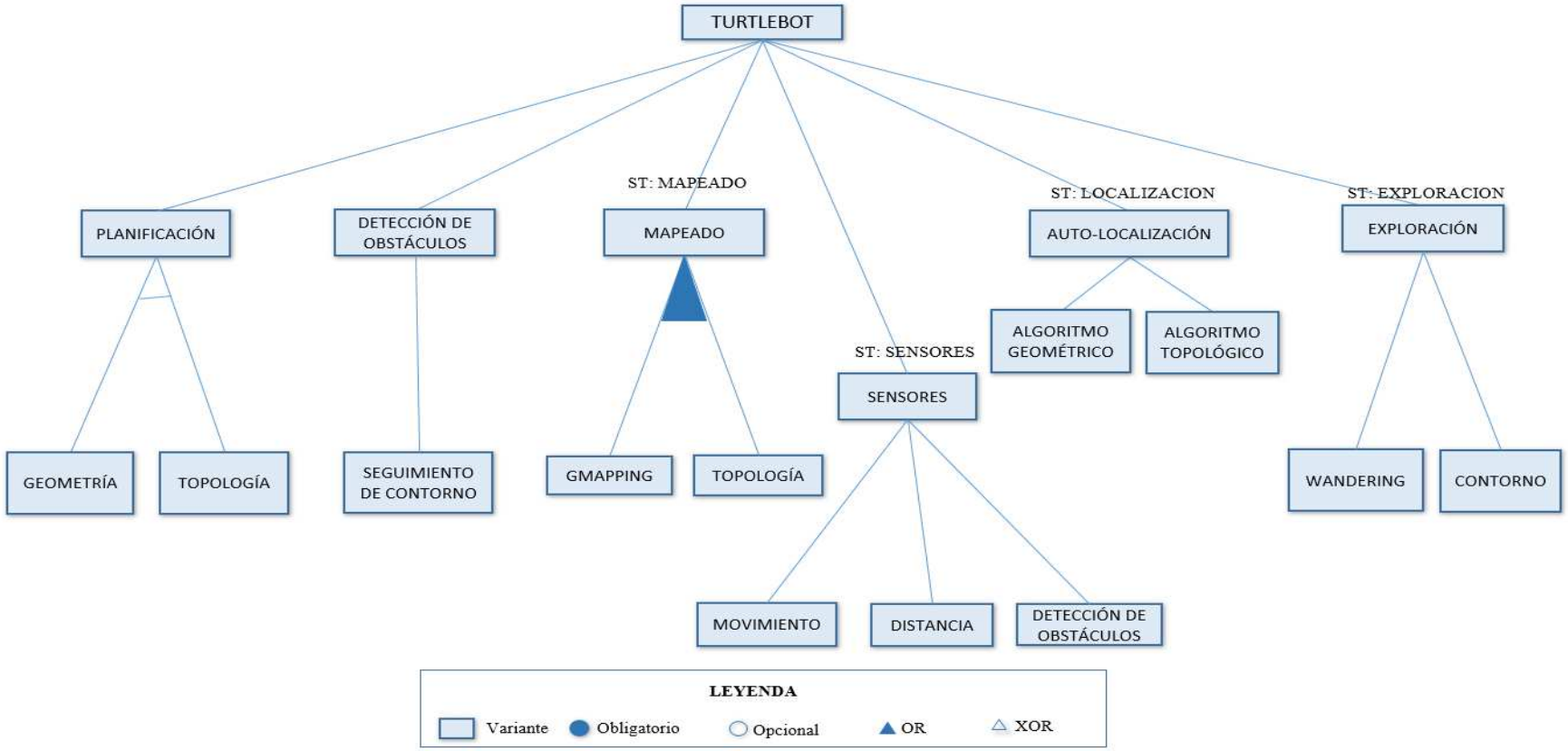
- Usos de sensores.
- Mapas de navegación.
- Detección y comunicación de límites y obstáculos.
- Diferentes algoritmos de exploración.



Experimentación

Parte 3
Simulación con robot TurtleBot

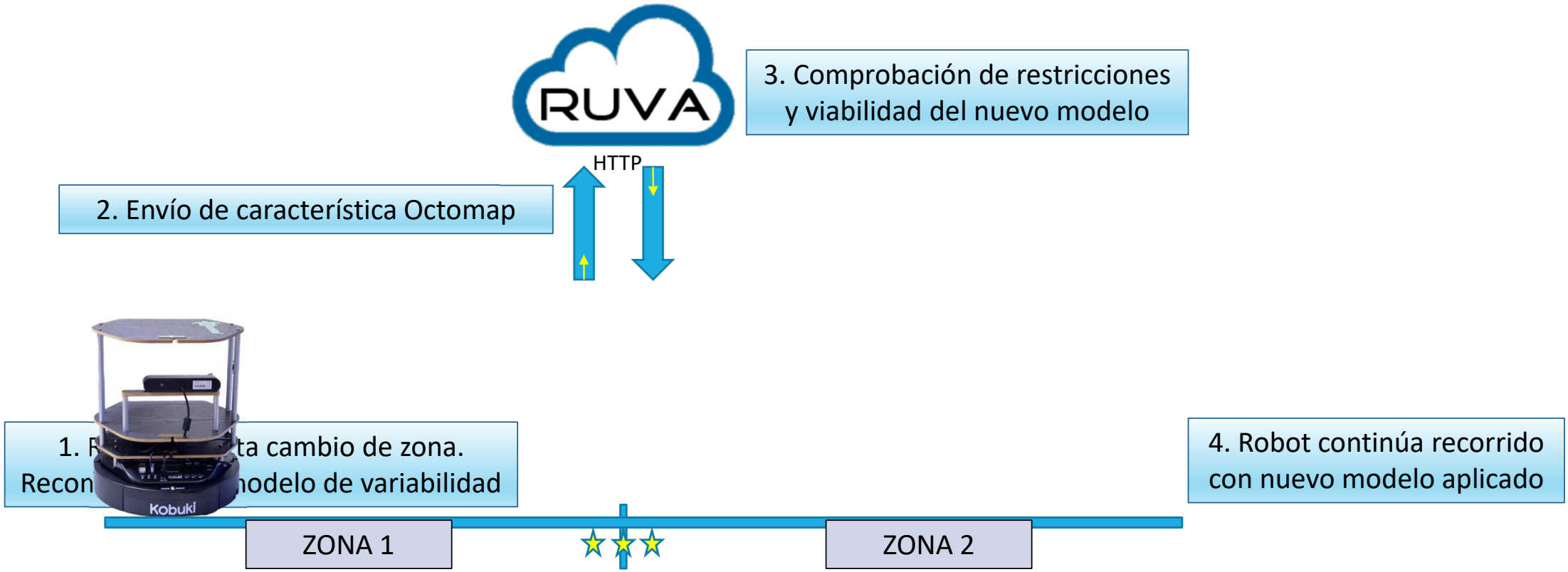
Modelo de variabilidad del TurtleBot



Experimentación

Parte 3
Simulación con robot TurtleBot

Desarrollo del experimento



Experimentación

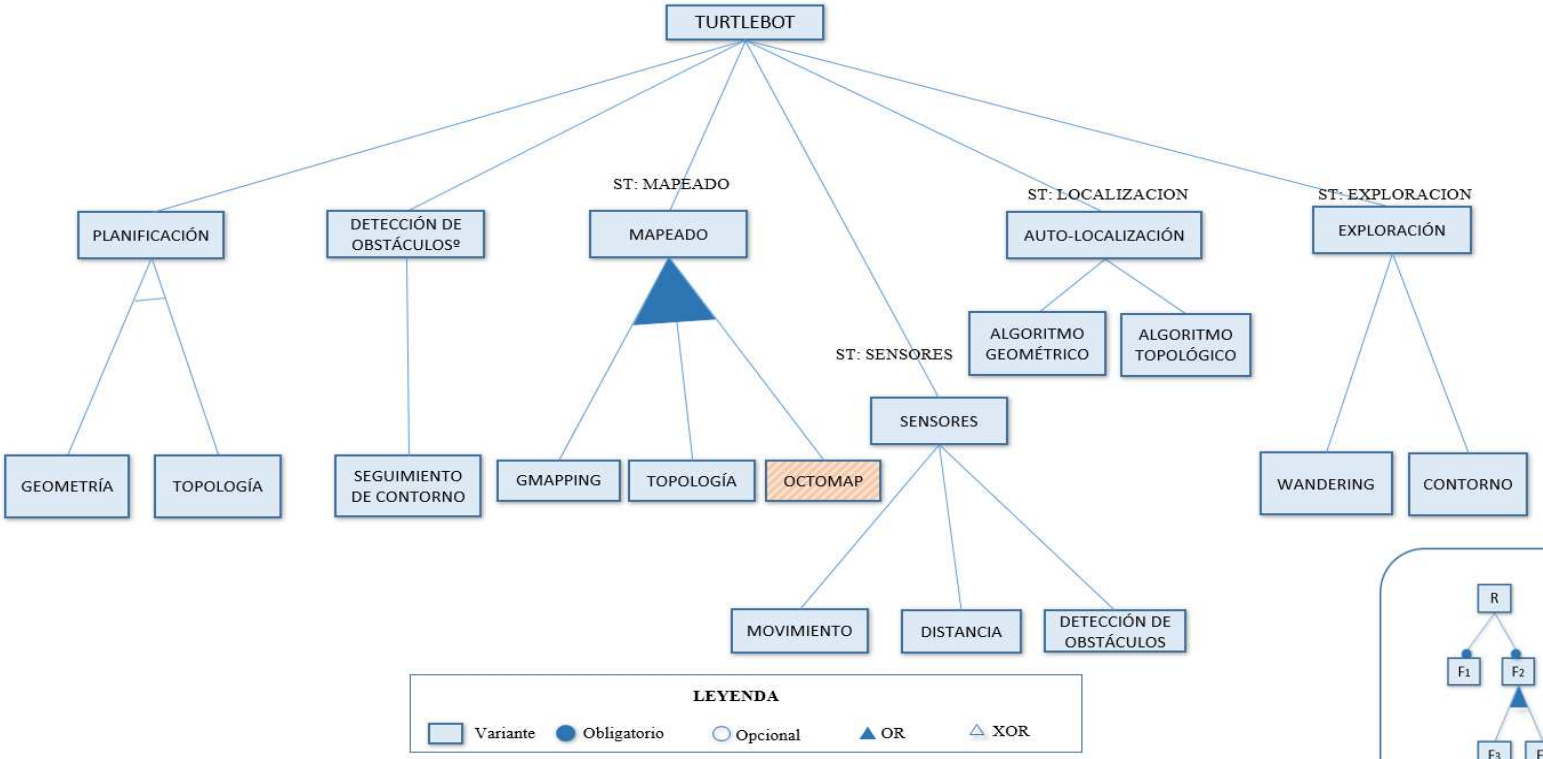
Parte 3
Simulación con robot TurtleBot

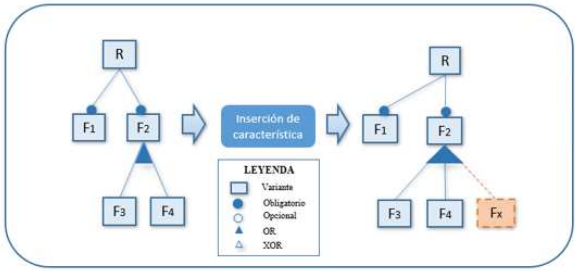
Desarrollo del experimento

Experimentación

Parte 3
Simulación con robot TurtleBot

Modelo de variabilidad resultante





Universidad Rey Juan Carlos

Experimentación

Conclusiones del experimento en pruebas de eficiencia y escalabilidad

- El algoritmo de variabilidad dinámica **se comporta en la forma prevista** insertando características localizadas con supertipos.
- RuVa se muestra bastante eficiente en la mayor parte de los casos, incluso con modelos de variabilidad grandes, donde el **tiempo de respuesta** se mantiene **bajo**.
- El modulo de optimización es capaz de **equilibrar** la inserción de nuevas características en caso de inserciones múltiples.

Experimentación

Conclusiones del experimento con el robot

- La integración de sistemas dependientes del contexto con nuestro algoritmo de variabilidad dinámica permite **reconfigurar la variabilidad estructural durante la ejecución del sistema**.
- La **definición** y **aplicación** de los **supertipos** actúa como limitante y **puede afectar al resultado final**. Una elección diferente de supertipos puede incrementar los resultados fallidos.
- **El experimento realizado en sentido inverso** (el diseñador añade una característica y el cambio se comunica al robot) **podría arrojar resultados distintos.**

Conclusiones

- Se proporciona una **solución para reconfigurar la variabilidad dinámica con comprobación de restricciones en tiempo de ejecución**, siendo este aspecto pobremente tratado en trabajos anteriores.
- El optimizador de características ayuda a **equilibrar la inserción en modelos de variabilidad grandes**, siendo este un aspecto novedoso frente a otras propuestas.
- La **comunicación del algoritmo con sistemas dependientes del contexto** se ha demostrado posible, asumiendo que la visualización del nuevo modelo de variabilidad no es una tarea crítica.
- La **gestión semiautomática de los puntos de variación** aporta un aspecto novedoso y avanza el estado del arte de otras propuestas para modelos de variabilidad extensibles y abiertos.

Futuras líneas de investigación

- **Integración de RuVa** con aproximaciones de Líneas de Producto Software Dinámicas (DSPL) y otros sistemas ciber-físicos.
- Investigación de **topologías de modelos de variabilidad más adecuados** que permitan una mejor distribución de supertipos para superar el 92% de eficacia.
- Investigar otros mecanismos para **añadir características sin el uso de supertipos predefinidos**.
- Explorar formas más **eficientes** de **comprobar las restricciones en tiempo de ejecución**.
- Explorar el uso de Cloud Robotics para la **actualización de software en robots** conectándolo con el algoritmo propuesto.

Artículos

Publicaciones derivadas de la tesis

- R. Capilla, A. Valdezate, J. C. Dueñas, **An analysis of variability modeling and management tools for product line development,** Software and Services Variability Management Workshop, Helsinki University of Technology (2007)
- R. Capilla, A. Valdezate, F. J. Díaz, **A Runtime Variability Mechanism Based on Supertypes,** FAS*W@SASO/ICCAC 2016: 6-11 (2016)
- A. Valdezate, R. Capilla, J. Crespo, R. Barber,**RuVa: A Runtime Software Variability Algorithm,** IEEE Access 10: 52525-52536 (2022)
- B. Caesar, A. Valdezate, J. Ladiges, R. Capilla, A. Fay,**A Process for Identifying & Modeling Reconfiguration Relevant System Context,** IEEE Transactions on Automation Science and Engineeering (2022). (en primera revisión)

Artículos

Otros publicaciones

- Jeffrey C. Carver, Eduardo Santana de Almeida, Rafael Capilla, Leandro L. Minku, Marco Torchiano, Alejandro Valdezate
  Empirical Software Engineering, Predictive Models, and Product Lines, IEEE Software. 35(3): 8-11 (2018)
- Jeffrey C. Carver, Rafael Capilla, Birgit Penzenstadler, Alexander Serebrenik, Alejandro Valdezate
  Gender, Sentiment and Emotions, and Safety-Critical Systems
  IEEE Software. 35(6): 16-19 (2018)
- Alejandro Valdezate, Rafael Capilla, Gregorio Robles, Víctor Salamanca
  Can instability variations warn developers when open-source projects boost? CoRR abs/2204.05209 (2022). Invitado a enviar a Empirical Software Engineering Journal, Springer.

Universidad Rey Juan Carlos

Defensa de Tesis Doctoral

Programa en Tecnologías de la Información y las Comunicaciones

Doctorando: Alejandro Valdezate Sánchez

Director: Dr. Rafael Capilla Sevilla

VARIABILIDAD DINÁMICA EN SISTEMAS SENSIBLES AL CONTEXTO

Gracias por su atención