Формализация системы типов OCL

<https://www.omg.org/spec/OCL/2.4/>

Спецификация

<https://www.researchgate.net/publication/266024091_The_Amsterdam_Manifesto_on_OCL>

Тут говорится, что OclAny должен быть супер-типом только для базовых типов

<https://www.brucker.ch/bibliography/download/2002/brucker.ea-proposal-2002.pdf>

Это вроде одна из первых статей с общими идеями про универсумы типов и т.п.

<https://www.brucker.ch/bibliography/download/2015/brucker.ea-formal-semantics-ocl-2.5-2015.pdf>

Проверить есть ли там кортежи, вроде только пары. Там типы определяются как типы Isabelle HOL. Типы не моделируются сами по себе, например, там не описывается отношение порядка для типов. Типы рассматриваются с точки зрения значений.

<https://www.brucker.ch/bibliography/download/2009/brucker.ea-ocl-null-2009.pdf>

Интересная статья про null

<https://www.eclipse.org/modeling/mdt/ocl/docs/publications/OCL2015SafeNavigation/SafeNavigation.pdf>

Ещё про null

<https://link.springer.com/content/pdf/10.1007%2F11494881_6.pdf>

<https://openaccess.leidenuniv.nl/bitstream/handle/1887/4362/03.pdf?sequence=23>

2005 Marcel Kyas. An Extended Type System for OCL Supporting Templates and Transformations

Рассматриваются некоторые проблемы системы типов OCL и предлагается её расширить путем добавления пересечений и объединений типов. В том числе вводится универсальный тип (аналог нашего SupType). Объединение типов определяется как least upper bound. Аналогично и у нас вычисляется объединение типов. Разница в том, что мы не определяем объединение как самостоятельный тип, а просто ищем наиболее подходящий тип из существующих.

В статье предлагается ещё 3 каких-то расширения, в которые сложно быстро вникнуть. Что-то про ограничения на типы, возможно у меня это реализовано. Приводятся ссылки на другие статьи, описывающие систему типов OCL.

Говорится про Adequacy and Decidability системы типов, нужно вникнуть что это за свойства и возможно доказать их для нашей системы.

<https://www.springer.com/la/book/9783319728131>

<https://books.google.ru/books?id=dZRIDwAAQBAJ&pg=PA8&lpg=PA8&dq=ocl+type+system&source=bl&ots=c2BOUQ1oK_&sig=tCPFtXIOPAp6ynA9tTF4fFanNfw&hl=ru&sa=X&ved=2ahUKEwjCu8-z25LfAhVvp4sKHWdkCw0Q6AEwBnoECAQQAQ#v=onepage&q=ocl%20type%20system&f=false>

2018 Przigoda, N., Wille, R., Przigoda, J., Drechsler, R. Automated Validation & Verification of UML/OCL Models Using Satisfiability Solvers

Глава A Formal Interpretation of UML/OCL

Тут есть какая-то претензия на формальность, но всё ограничивается какими-то кортежными формулами, не формулируется и не доказывается ни одной теоремы.

<https://www.researchgate.net/publication/313070735_A_Dynamic_Type_System_for_OCL>

На мой взгляд рассматриваются какие-то надуманные проблемы, для решения которых нужно добавить динамическую типизацию. Определяются какие-то операции для типов, в частности smallestCommonSuperType, который очень похож на наше объединение типов. Это показывает, что такое объединение нужно не только нам. Вроде они реализовали всё это в платформе Neptune и скорее всего императивно. У нас же преимущество в том, что описание системы типов полностью формальное с доказательством всех теорем.

<http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=BA87BB56954BB0ADED83C6825F2D7B78?doi=10.1.1.31.6712&rep=rep1&type=pdf>

2000 Thomas Baar und Reiner Hahnle. An Integrated Metamodel for OCL Types

Это попытка формализовать систему типов OCL путем описания их метамодели. В спецификации OCL тоже в основном используется этот подход, упоминаются мета-типы и т.п. Проблема в том, что такой подход не позволяет, например, формально доказать, что система типов ациклична. Не позволяет формально определить наследование типов или вычисление супер-типа. Видимо, поэтому в спецификации OCL есть приложение, в котором всё это попытались описать формально. Но получилось всё равно не очень.

<https://www.researchgate.net/publication/220059427_OCL_145_vs_20_Expressions_Formal_semantics_and_expressiveness>

2004 Mar ́ıa Victoria Cengarle, Alexander Knapp. OCL 1.4/5 vs. 2.0 Expressions. Formal semantics and expressiveness

Описывается система вывода типов, операционная и денотационная семантика OCL. Очень хорошая статья, в ней говорится, что система типов образует как минимум частичный порядок. Говорится, что про парадоксы, связанные с Set(OclAny) < OclAny. Говорится про least upper bound и про то, что он может не существовать. Но это на наш взгляд как-раз проблема. В нашей работе мы добавляем SupType и делаем систему типов верхней полурешеткой. На самом деле это полная решетка и это легко доказать, но …

<https://www.researchgate.net/publication/220868389_A_Formal_Semantics_for_OCL_14>

María Victoria Cengarle, Alexander Knapp. A Formal Semantics for OCL 1.4

Ещё одна статья от этих же авторов, очень похожа на предыдущую. Достаточно подробно описана система типов.

<https://pdfs.semanticscholar.org/341b/f42c7f4c6a3c91dbfc32dfefa0b9ae43ba87.pdf>

1999 Tony Clark. Typechecking UML Static Models

В целом говорится про семантику, систему типов, есть математика. Но слишком в целом.

<https://www.researchgate.net/publication/221594522_A_New_Type_Checking_Approach_for_OCL_Version_20>

<https://books.google.ru/books?id=XalqCQAAQBAJ&pg=PA23&lpg=PA23&dq=A+new+type+checking+approach+for+OCL+2.0&source=bl&ots=4AMNXaLlt7&sig=XZ_GD3pwujces1HILZKM7qSOpms&hl=ru&sa=X&ved=2ahUKEwi_r4HC9JLfAhUjlosKHUJoBiAQ6AEwBHoECAYQAQ#v=onepage&q=A%20new%20type%20checking%20approach%20for%20OCL%202.0&f=false>

2002 Sch¨urr, A.: A new type checking approach for OCL 2.0?

Очень интересная статья с описанием каких-то проблем в системе типов OCL. Как я понял, в основном они связаны со множественным наследованием. Говорится про разные подходы к проверке типов, в том числе говорится про решетки, про smallest common supertype. Есть только фрагменты статьи, поэтому сложно понять, что они в итоге предлагают. Как я понимаю, мы множественное наследование вообще не рассматриваем, оно вполне может быть, ограничений на это нет. Проблема может быть в том, что в нашей системы может быть выведен слишком общий тип.

<https://ac.els-cdn.com/S1571066104051163/1-s2.0-S1571066104051163-main.pdf?_tid=9ed4f706-4caf-4c11-ab09-0b64fea9350b&acdnat=1544361218_397334f1a2ae349a8d32f1f3f729358d>

2004 Stephan Flake. OclType – A Type or Metatype ?

Что про powertype и т.п. Пока не очень понял, что за проблема там решается. Нужно подумать, что такое OclType в нашей теории.

<https://pdfs.semanticscholar.org/84a7/20c2a5c525819beb107364d53fb09f2be940.pdf>

2004. Stephan Flake. Towards the Completion of the Formal Semantics of OCL 2.0

Ощущение, что эта статья в итоге попала в спецификацию OCL. Проблема в том, что там в основном кортежи и всё как-то полуформально.

<http://santos.cs.ksu.edu/771-Distribution/Reading/Richters_2001_OCL.pdf>

2001 Richters, M., Gogolla, M.: OCL: Syntax, semantics, and tools

Или эта статья попала в спецификацию. Такие же кортежи, не слишком формально.

<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.58.9478&rep=rep1&type=pdf>

Stephan Flake and Wolfgang Mueller. An ASM Definition of the Dynamic OCL 2.0 Semantics

Аналогично, очень похоже на приложение к спецификации OCL, но в основном про объектную модель, мало про типы.

<https://d-nb.info/1136395199/04>

2017 Nils Przigoda. SMT-based Validation &; Verification of UML/OCL Models

Интересная диссертация. Вроде они транслируют UML+OCL во что-то.

<https://beyermatthias.de/papers/2017/Benefits_of_type_constraints_over_ocl_constraints.pdf>

2017 Matthias Beyer and Julian Ganz and Peter Wursthorn. Benefits of type constraints over OCL constraints

Как я понял, они преобразуют OCL выражения в какие-то ограничения для типов на языке Rust. И говорят, что это дает кучу преимуществ, а OCL вообще не нужен. Что-то есть в этой статье, но немного сомнительная.

<http://www.eecs.qmul.ac.uk/~ddino/papers/puml2k.pdf>

2000 Dino Distefano, Joost-Pieter Katoen and Arend Rensink. Towards model checking OCL

Тут определяется какой-то новый язык BOTL, в который транслируется OCL. Не подходит