Is UML practical? [closed]

Asked 16 years, 4 months ago Modified 5 years, 3 months ago

Viewed 35k times

127

As it currently stands, this question is not a good fit for our Q&A format. We expect answers to be supported by facts, references, or expertise, but this question will likely solicit debate, arguments, polling, or extended discussion. If you feel that this question can be improved and possibly reopened, visit the help center for guidance.

Closed 12 years ago.

In college I've had numerous design and UML oriented courses, and I recognize that UML can be used to benefit a software project, especially use-case mapping, but is it really practical? I've done a few co-op work terms, and it appears that UML is not used heavily in the industry. Is it worth the time during a project to create UML diagrams? Also, I find that class diagrams are generally not useful, because it's just faster to look at the header file for a class. Specifically which diagrams are the most useful?

Edit: My experience is limited to small, under 10 developer projects.

Edit: Many good answers, and though not the most verbose, I belive the one selected is the most balanced.

uml class-design diagram

Share

Improve this question

Follow

edited Aug 23, 2008 at 17:55

community wiki
7 revs
Chris

4 Results from a 2013 survey reveal it's not used as much as software engineering professors expect (!) and reveal some reasons why: oro.open.ac.uk/35805/8/UML%20in%20practice%208.pdf – Fuhrmanator Jun 11, 2015 at 23:21

31 Answers

Sorted by: Highest score (default)

1 2 Next

92

Using UML is like looking at your feet as you walk. It's making conscious and explicit something that you can usually do unconsciously. Beginners need to think carefully about what they're doing, but a professional programmer already knows what they're doing. Most of the time, writing the code itself is quicker and more effective than writing about the code,

> because their programming intuition is tuned to the task.

It's not just about what you're doing though. What about the new hire who comes in six months from now and needs to come up to speed on the code? What about five years from now when everyone currently working on the project is gone?

It's incredibly helpful to have some basic up to date documentation available for anyone who joins the project later. I don't advocate full blown UML diagrams with method names and parameters (WAY too difficult to maintain), but I do think that a basic diagram of the components in the system with their relationships and basic behavior is invaluable. Unless the design of the system changes drastically, this information shouldn't change a lot even as the implementation is tweaked.

I've found that the key to documentation is moderation. No one is going to read 50 pages of full blown UML diagrams with design documentation without falling asleep a few pages in. On the other hand, most people would love to get 5-10 pages of simple class diagrams with some basic descriptions of how the system is put together.

The other case where I've found UML to be useful is for when a senior developer is responsible for designing a component but then hands the design to a junior developer to implement.

Share Improve this answer Follow

edited Aug 21, 2008 at 20:00

community wiki
2 revs
17 of 26

35 "What about the new hire who comes in six months from now and needs to come up to speed on the code?" Errr.. how about looking at the code? That is the only accurate and complete way of getting up to speed on the code. Also the most natural way, considering we are programmers. I consider the notion that we have to look at a diagram to understand code completely laughable and also depressing that this garbage somehow became widespread. – BobTurbo Jun 3, 2011 at 8:31

8 Agree with BobTurbo, I've never had any use for UML, especially somebody else's UML. I always prefer to go straight to the code. – James Adam Feb 20, 2013 at 19:38

10 I have found drawing a UML class diagram before embarking on coding has saved me time. It has allowed me to visualise design faults and flaws and to discuss these with my colleagues. Better still if they are drawn using CASE tools, they will generate the basic structural code of your app. So the payback is three fold. – Andrew S Jan 28, 2014 at 7:09

16 A picture is still worth a thousand words, even when it's code, @BobTurbo. I don't see any rational argument against this -- and that includes arguments that begin with "Well *real* programmers..." If I'm going to have a conversation about architecture with my team, I'm not going to scotch tape 10 pages of source code onto a whiteboard. – user201891 Jun 19, 2015 at 21:51

4 @BobTurbo But is it always easy though reading another person's code? How confident are you just reading Microsoft's .NET source code and understanding it all in 1 month? Even if you do understand, you're always left with the question, "Why did he do it this way? Why not that way?" Diagrams also allows proper planning and distribution of coding tasks so that you do not overlap tasks. It also helps in determining the feasibility, cost as well as proof to justify your decision to the client/superiors/non-coders. In essence, UML is recommended unless you're developing a hello world app. – John Evans Solachuk Dec 5, 2016 at 4:06

50

In a sufficiently **complex system** there are some places where some `UML` is considered useful.

The useful diagrams for a system, vary by applicability. But the most widely used ones are:

- Class Diagrams
- State Diagrams
- Activity Diagrams
- Sequence Diagrams

There are many enterprises who swear by them and many who outright reject them as an utter waste of time and effort.

It's best not to go overboard and think what's best for the project you are on and pick the stuff that is applicable and makes sense.

Share Improve this answer Follow

edited Sep 7, 2019 at 4:08

community wiki
3 revs, 2 users 50%
Pascal

38

Using UML is like looking at your feet as you walk. It's making conscious and explicit something that you can usually do unconsciously. Beginners need to think carefully about what they're doing, but a professional programmer already knows what they're doing. Most of the time, writing the code itself is quicker and more effective than writing about the code, because their programming intuition is tuned to the task.

The exception is why you find yourself in the woods at night without a torch and it's started to rain - then you need to look at your feet to avoid falling down. There are times when the task you've taken on is more complicated than your intuition can handle, and you need to slow down and state the structure of your program explicitly. Then UML is one of many tools you can use. Others include pseudocode, high-level architecture diagrams and strange metaphors.

Share Improve this answer Follow

answered Aug 21, 2008 at 0:09

community wiki

Marcus Downing

4 Weird thing about this website, there's no way to tell that you're quoting somebody in the first paragraph and then disagreeing with them.. but yeah, true: pseudocode is also a great diagramming technique, and very commonly used. Strange metaphors involving woods, torches, etc. are great too. – Dan Rosenstark Oct 22, 2008 at 6:13

not agree at all - if you make complex components - uml is must – yehonatan yehezkel May 26, 2019 at 12:19

21

Generic work-flow and DFDs can be very useful for complex processes. All other diagramming (ESPECIALLY UML) has, in my experience, without exception been a painful waste of time and effort.

Share Improve this answer Follow

answered Aug 20, 2008 at 20:56

community wiki
Stu

16

I'd have to disagree, UML is used all over the place - anywhere a IT project is being designed UML will usually be there.

Now whether it is being used *well* is another matter.

As Stu said, I find both Use Cases (along with the use case descriptions) and activity diagrams to be the most

helpful from a developer point of view.

Class diagram can be very useful when trying to show relationships, as well as object attributes, such as persistence. When it comes to adding ever single attribute or property they are usually overkill, especially as they often become out of date quickly once code is written.

One of the biggest problems with UML is the amount of work required to keep it up to date once code is being generated, as there are few tools that can re-engineer UML from code, and few still that do it well.

Share Improve this answer Follow

answered Aug 20, 2008 at 21:02

community wiki
samjudson

14

I will qualify my answer by mentioning that I don't have experience in large (IBM-like) corporate development environments.

The way I view UML and the Rational Unified Process is that it's more *TALKING* about what you're going to do than actually *DOING* what you're going to do.

(In other words it's largely a waste of time)

Share Improve this answer

answered Aug 20, 2008 at 21:10

Follow

community wiki
Seibar

10 I won't vote you down because I think it's a good answer, but I couldn't disagree more. Every time I diagram something I save myself hours or months of dev time, and I almost always develop alone (recently). – Dan Rosenstark Oct 22, 2008 at 6:09

4 Talking and writing about what you want to do helps you and other to understand and catch possible issues sooner. – Kamran Bigdely Jun 16, 2015 at 17:34

I voted down because in any organization more than one person talking is definitely not a waste of time. – Eric Nov 29, 2021 at 20:50

14

Throw away only in my opinion. UML is a great tool for communicating ideas, the only issue is when you store and maintain it because you are essentially creating two copies of the same information and this is where it usually blows. After the initial round of implementation most of the UML should be generated from the source code else it will go out of date very quickly or require a lot of time (with manual errors) to keep up to date.

Share Improve this answer

answered Aug 21, 2008 at 19:54

Follow

community wiki

Quibblesome

Wow. What about a tool that maintains the diagram in sync with the code and vice versa, like Together?
– Dan Rosenstark Oct 22, 2008 at 6:08

1 The fact that UML can be generated from the source means, to me, that there's no benefit in reading the UML over just reading the source. In other words it's a waste.
– Marcus Downing Sep 25, 2011 at 23:42

2 @MarcusDowning No, it helps new hires massively. It's quicker to look at UML than the code. – Kamran Bigdely Jun 16, 2015 at 17:37

10

I co-taught a senior-level development project course my last two semesters in school. The project was intended to be used in a production environment with local non-profits as paying clients. We had to be certain that code did what we expected it to and that the students were capturing all the data necessary to meet the clients' needs.

Class time was limited, as was my time outside of the classroom. As such, we had to perform code reviews at every class meeting, but with 25 students enrolled individual review time was very short. The tool we found most valuable in these review sessions were ERD's, class diagrams and sequence diagrams. ERD's and class diagrams were done only in Visual Studio, so the time required to create them was trivial for the students.

The diagrams communicated a great deal of information very quickly. By having a quick overview of the students'

designs, we could quickly isolate problem areas in their code and perform a more detailed review on the spot.

Without using diagrams, we would have had to take the time to go one by one through the students' code files looking for problems.

Share Improve this answer Follow

answered Aug 20, 2008 at 22:57

community wiki
Jason Sparks

+1 Good visualisation of data helps to expose issues and trends otherwise obscured with unrelevant detail.
– Vlad Gudim Jan 14, 2010 at 11:43

8

I am coming to this topic a little late and will just try an clarify a couple minor points. Asking if UML is useful as far too broad. Most people seemed to answer the question from the typical/popular UML as a drawing/communication tool perspective. Note: Martin Fowler and other UML book authors feel UML is best used for communication only. However, there are many other uses for UML. Above all, UML is a modeling language that has notation and diagrams mapped to the logical concepts. Here are some uses for UML:

- Communication
- Standardized Design/Solution documentation

- DSL (Domain Specific Language) Definition
- Model Definition (UML Profiles)
- Pattern/Asset Usage
- Code Generation
- Model to Model transformations

Given the uses list above the posting by Pascal is not sufficient as it only speaks to diagram creation. A project could benefit from UML if any of the above are critical success factors or are problem areas that need a standardized solution.

The discussion should expanded out from how UML can be over kill or applied to small projects to discuss when UML makes sense or will actually improve the product/solution as that is when UML should be used. There are situations where UML for one developer could sense as well, such as Pattern Application or Code Generation.

Share Improve this answer Follow

edited Nov 15, 2016 at 10:59

community wiki
3 revs, 3 users 93%
Ted Johnson

UML has worked for me for years. When I started out I read Fowler's UML Distilled where he says "do enough

6

modelling/architecture/etc.". Just use what you *need*!

Share Improve this answer Follow

answered Aug 24, 2008 at 9:41

community wiki
Sean Kearon

5

Though this discussion has long been inactive, I have a couple of -to my mind important- points to add.

Buggy code is one thing. Left to drift downstream, design mistakes can get **very** bloated and ugly indeed. UML, however, is self-validating. By that I mean that in allowing you to explore your models in multiple, mathematically closed and mutually-checking dimensions, it engenders robust design.

UML has another important aspect: it "talks" directly to our strongest capability, that of visualisation. Had, for example, ITIL V3 (at heart simple enough) been communicated in the form of UML diagrams, it could have been published on a few dozen A3 foldouts. Instead, it came out in several tomes of truly biblical proportions, spawning an entire industry, breathtaking costs and widespread catatonic shock.

Share Improve this answer Follow

answered May 13, 2011 at 10:18

community wiki
Thug

2 Have you read the UML standard? it has NO mathematical background whatsoever, and it even has internal inconsistencies. It is also very difficult to understand: for example, rounded rectangles are used for two completely different things (states in state machines and actions and activities in activity diagrams). It is awful. – vainolo Jun 27, 2012 at 20:48

4

From a QA Engineer's perspective, UML diagrams point out potential flaws in logic and thought. Makes my job easier :)

Share Improve this answer Follow

answered Oct 22, 2008 at 6:12

community wiki
Nick Stinemates

3

I believe there may be a way to utilize Cockburn style UML fish,kite, and sea-level use cases as described by Fowler in his book "UML Distilled." My idea was to employ Cockburn use cases as an aid for code readability.

So I did an experiment and there is a post here about it with the Tag "UML" or "FOWLER." It was a simple idea for c#. Find a way to embed Cockburn use cases into the

namespaces of programming constructs (such as the class and inner class namespaces or by making use of the namespaces for enumerations). I believe this could be a viable and simple technique but still have questions and need others to check it out. It could be good for simple programs that need a kind of pseudo-Domain Specific Language which can exist right in the midst of the c# code without any language extensions.

Please check out the post if you are interested. Go here.

Share Improve this answer Follow

edited May 23, 2017 at 12:10

community wiki
2 revs
fooledbyprimes

3

I think the UML is useful thought I think the 2.0 spec has made what was once a clear specification somewhat bloated and cumbersome. I do agree with the edition of timing diagrams etc since they filled a void...

Learning to use the UML effectively takes a bit of practice. The most important point is to communicate clearly, model when needed and model as a team. Whiteboards are the best tool that I've found. I have not seen any "digital whiteboard software" that has managed to capture the utility of an actual whiteboard.

That being said I do like the following UML tools:

1. Violet - If it were any more simple it would be a piece of paper
2. Altova UModel - Good tool for Java and C# Modeling
3. MagicDraw - My favorite commercial tool for Modeling
4. Poseidon - Decent tool with good bang for the buck
5. StarUML - Best open source modeling tool

Share Improve this answer Follow

answered Sep 25, 2008 at 5:09

community wiki
Daniel Honig

3

UML diagrams are useful for capturing and communicating requirements and ensuring that the system meets those requirements. They can be used iteratively and during various stages of planning, design, development, and testing.

From the topic: **Using Models within the Development Process** at http://msdn.microsoft.com/en-us/library/dd409423%28VS.100%29.aspx

> A model can help you visualize the world in which your system works, clarify users' needs, define the architecture of your system, analyze

> the code, and ensure that your code meets the requirements.

You might also want to read my response to the following post:

How to learn “good software design/architecture”? at https://stackoverflow.com/questions/268231/how-to-learn-good-software-design-architecture/2293489#2293489

Share Improve this answer Follow

edited May 23, 2017 at 12:26

community wiki
2 revs
Esther Fan - MSFT

2

I see sequence diagrams and activity diagrams used fairly often. I do a lot of work with "real-time" and embedded systems that interact with other systems, and sequence diagrams are very helpful in visualizing all the interactions.

I like to do use-case diagrams, but I haven't met too many people who think they are valuable.

I've often wondered whether Rational Rose is a good example of the kinds of applications you get from UML-model-based design. It's bloated, buggy, slow, ugly, ...

Share Improve this answer Follow

answered Aug 20, 2008 at 21:02

community wiki
Kristopher Johnson

2

I found UML not really useful for very small projects, but really suitable for larger ones.

Essentially, it does not really matter what you use, you just have to keep two things in mind:

- You want some sort of architecture planning
- You want to be sure that everyone in the team is actually using the same technology for project planning

So UML is just that: A standard on how you plan your projects. If you hire new people, there are more likely to know any existing standard - be it UML, Flowchard, Nassi-Schneiderman, whatever - rather than your exising in-house stuff.

Using UML for a single developer and/or a simple software project seems overkill to me, but when working in a larger team, I would definitely want some standard for planning software.

Share Improve this answer Follow

answered Aug 20, 2008 at 23:03

community wiki
Michael Stum

2

UML is useful, yes indeed! The main uses I've made of it were:

- Brainstorming about the ways a piece of software should work. It makes easy to communicate what you are thinking.
- Documenting the architecture of a system, it's patterns and the main relationships of its classes. It helps when someone enters your team, when you're leaving and want to make sure your successor will understand it, and when you eventually forget what the hell that little class was meant for.
- Documenting any architectural pattern you use on all your systems, for the same reasons of the dot above

I only disagree with Michael when he says that *using UML for a single developer and/or a simple software project seems overkill* to him. I've used it on my small personal projects, and having them documented using UML saved me a lot of time when I came back to them seven months later and had completely forgotten how I had built and put together all those classes.

Share Improve this answer

Follow

edited May 23, 2017 at 12:02

community wiki
2 revs
Mario Marinato -br-

2

One of the problems I have with UML is the understandability of the specification. When I try to really understand the semantics of a particular diagram I quickly get lost in the maze of meta-models and meta-meta-models. One of the selling points of UML is that it is less ambiguous than natural language. However, if two, or more, engineers interpret a diagram differently, it fails at the goal.

Also, I've tried asking specific questions about the super-structure document on several UML forums, and to members of the OMG itself, with little or no results. I don't think the UML community is mature enough yet to support itself.

Share Improve this answer

Follow

answered Sep 25, 2008 at 5:04

community wiki
Trent

Coming from a student, I find that UML has very little use. I find it ironic that PROGAMERS have yet to develop a

2

program that will automatically generate the things that you have said are necessary. It would be extremely simple to design a feature into Visual Studio that could pull pieces of the data, seek for definitions, and product answers sufficent so that anyone could look at it, great or small, and understand the program. This would also keep it up to date because it would take the information directly from the code to produce the information.

Share Improve this answer Follow

answered Jan 27, 2009 at 5:01

community wiki
Mike

> It's not that easy! If you use reverse engineering to extract a UML model from real code, you tend to end up with so much detail in your UML model that's it useless. No doubt this is being pursued by researchers, but it's not an easy problem to solve. – ComDubh Feb 4, 2016 at 17:20

2

UML is used as soon as you represent a class with its fields and methods though it's just a kind of UML diagram.

The problem with UML is that the founders book is too vague.

UML is just a language, it's not really a method.

As for me, I really find annoying the lack of UML schema for Opensource Projects. Take something like Wordpress,

you just have a database schema, nothing else. You have to wander around the codex api to try to get the big picture.

Share Improve this answer Follow

answered Aug 9, 2009 at 10:17

community wiki
programmernovice

1

UML has its place. It becomes increasingly important as the size of the project grows. If you have a long running project, then it is best to document everything in UML.

Share Improve this answer Follow

answered Aug 20, 2008 at 20:58

community wiki
Vaibhav

> Or at the very least the key design issues. – Silvercode Nov 6, 2008 at 8:20

1

UML seems to good for large projects with large teams of people. However I've worked in small teams where communication is better.

Using UML-esque diagrams is good though, especially in the planning stage. I tend to think in code, so I find writing

large specs hard. I prefer to write down the inputs' and outputs' and leave the developers to design the bit in the middle.

Share Improve this answer Follow

answered Aug 21, 2008 at 5:15

community wiki
graham.reeds

1 Also in a bit smaller projects it is frustrating to work by communicating in my opinion. If you have to ask and tell all the time lots of stuff, it interrupts work and no documents are produced. Instead the key design principles should be documented and modeled. – Silvercode Nov 6, 2008 at 8:24

1

I believe UML is useful just for the fact that it gets people to think about the relationships between their classes. It is a good starting point to start thinking about such relationships, but it is definitely not a solution for everybody.

My belief is that the use of UML is subjective to the situation in which the development team is working.

Share Improve this answer Follow

answered Aug 21, 2008 at 19:50

community wiki
Adam Mika

0

In my experience:

The ability to create and communicate meaningful code diagrams is a necessary skill for any software engineer who is developing new code, or attempting to understand existing code.

Knowing the specifics of UML - when to use a dashed line, or a circle endpoint - is not quite as necessary, but is still good to have.

Share Improve this answer Follow

answered Aug 21, 2008 at 19:29

community wiki

David Koelle

0

UML is useful in two ways:

- Technical side: a lot of people (manager and some functional analyst) think that UML is a luxury feature because *The code is the documentation: you start coding, after you debug and fix*. The sync of UML diagrams with code and analisys force you to understand well the requests of the customer;
- Management side: the UMl diagrams are a mirror of the requires of the customer who is inaccurate: if you code without UML, maybe you can find a bug in requires after a lot of hours of work. The diagrams UML allow you to find the possible controversal points and to resolve before the coding =>help your planning.

Generally, all the projects without UML diagrams have a superficial analysis or they have short size.

if you're in linkedin group **SYSTEMS ENGINEERS**, see my old discussion.

Share Improve this answer Follow

answered Jan 13, 2009 at 14:19

community wiki
alepuzio

-1

UML is definitely helpful just as junit is essential. It all depends how you sell the idea. Your program will work without UML just as it would work without unit tests. Having said that, you should create do UML as along it is connected to your code, i.e when you update UML diagrams it updates your code, or when you update your code it auto generates the UML. Don't do just for the sake of doing it.

Share Improve this answer Follow

answered Feb 20, 2010 at 3:20

community wiki
fastcodejava

-2

UML definetly has its place in the industry. Imagine you are building software for Boing aircraft or some other complex system. UML and RUP would be great help here.

Share Improve this answer Follow

answered Apr 13, 2009 at 21:18

community wiki
EdinD

In the end UML only exist because of RUP. Do we need UML or any of its related stuff to use Java/.Net ? The

-2

practical answer is they have their own documenation (javadoc etc) which is sufficient and lets us get our job done!

UML no thanx.

Share Improve this answer Follow

answered Apr 14, 2009 at 4:27

community wiki
mP.

RUP is about Process Management, UML is about Language. UML is usefull when you deal with a lot of people and need a common language. – programmernovice Aug 9, 2009 at 10:20

1 Ever heard of chinese whispiers - the more translating one does from one form to another meaning, difference and error creeps in. If UML is so good why dont Microsoft, Sun, Google include UML in their product details ? You will have a hard time finding any. Whatever happened to the tooling ? Forward/backward two way tools ?They dont exist because that fad died because of lack of merit. – mP. Aug 11, 2009 at 11:55

UML is just one of methods for communication within people. Whiteboard is better.

-3

Share Improve this answer Follow

answered Aug 20, 2008 at 21:02

community wiki
popopome