

1 Title 3

2 ANNE-MARIE ROMMERDAHL, SDU, Denmark

3 JEREMY ALEXANDER RAMÍREZ GALEOTTI, SDU, Denmark

4 DIMITRIOS DAFNIS, SDU, Denmark

5 NASIFA AKTER, SDU, Denmark

6 MOHAMMAD HOSEIN KARDOUNI, SDU, Denmark

7 BEN TROVATO* and **G.K.M. TOBIN***, Institute for Clarity in Documentation, USA

8 LARS THØRVÄLD, The Thørväld Group, Iceland

9 VALERIE BÉRANGER, Inria Paris-Rocquencourt, France

10 A clear and well-documented \LaTeX document is presented as an article formatted for publication by ACM in a conference proceedings or journal publication. Based on the “acmart” document class, this article presents and explains many of the common variations, as well as many of the formatting elements an author may use in the preparation of the documentation of their work.

11 CCS Concepts: • **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

12 Additional Key Words and Phrases: Do, Not, Use, This, Code, Put, the, Correct, Terms, for, Your, Paper

13 ACM Reference Format:

14 Anne-Marie Rommerdahl, Jeremy Alexander Ramírez Galeotti, Dimitrios Dafnis, Nasifa Akter, Mohammad Hosein Kardouni, Ben Trovato, G.K.M. Tobin, Lars Thørväld, and Valerie Béranger. 2018. Title 3. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 16 pages. <https://doi.org/XXXXXXX>.

15 1 Introduction

16 ACM’s consolidated article template, introduced in 2017, provides a consistent \LaTeX style for use across ACM publications, and incorporates accessibility and metadata-extraction functionality necessary for future Digital Library endeavors. Numerous ACM and SIG-specific \LaTeX templates have been examined, and their unique features incorporated into this single new template.

17*Both authors contributed equally to this research.

18 Authors’ Contact Information: Anne-Marie Rommerdahl, SDU, Odense, Denmark, anrom25@student.sdu.dk; Jeremy Alexander Ramírez Galeotti, SDU, Odense, Denmark, jeram25@student.sdu.dk; Dimitrios Dafnis, SDU, Odense, Denmark, didaf25@student.sdu.dk; Nasifa Akter, SDU, Copenhagen, Denmark, naakt23@student.sdu.dk; Mohammad Hosein Kardouni, SDU, Odense, Denmark, mokar25@student.sdu.dk; **Ben Trovato**, trovato@corporation.com; G.K.M. Tobin, webmaster@marysville-ohio.com, Institute for Clarity in Documentation, Dublin, Ohio, USA; Lars Thørväld, The Thørväld Group, Hekla, Iceland, larst@affiliation.org; Valerie Béranger, Inria Paris-Rocquencourt, Rocquencourt, France.

19 Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

20 © 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

21 Manuscript submitted to ACM

22 Manuscript submitted to ACM

53 2 Background and Related Work

54 Software reuse is a broad term, that refers to the practice of reusing previously written code, rather than coding from
 55 scratch. It is such an important part of software engineering, that one of the ways to measure the quality of software is
 56 by it's 'Reusability'[9] - i.e. the degree to which the application or its components can be reused. There are multiple
 57 benefits to practicing reuse in software engineering. One developer could save time by using another developer's
 58 reusable component, rather than coding their own. The developer avoids both the work of writing the syntax and
 59 designing the logic of the component. The developer can design their own reusable components, keeping all the logic
 60 in one place, which can then be tested thoroughly. However, despite reuse being an important practice in software
 61 engineering, there is still a limited focus on this practice when it comes to low-code development platforms (LCDP).

62 A study from 2021 studied several low-code platforms (LCPs), in order to identify characteristic features of LCPs.
 63 The identified features were presented according to how frequent they occurred, with domain-specific reference artifacts
 64 being categorized as 'rare'. Most studied systems offered catalogs of "reusable functions or examples of predefined
 65 processes", but they were found to be generic, or have a limited scope[10]. This lack of focus on promoting reuse may
 66 impact the so-called 'Citizen Developers', who have little or no coding knowledge, and whom may then miss out on the
 67 benefits of reuse.

68 There have been proposed some ideas on how to promote reuse for LCPs, such as the strongly-typed rich templating
 69 language OSTRICH, developed for the model-driven low-code platform OutSystems[32]. OutSystems provides scaffolding
 70 mechanisms for common development patterns and sample screen templates, both designed by experts on
 71 domain-specific languages (DSL). The practice of using templates in the OutSystems platform involves cloning and
 72 modifying samples, which may require more knowledge than the end-user possesses. The goal of OSTRICH is to remove
 73 this need for adaptation when using templates, to remove the knowledge-barrier when making use of the available
 74 templates. This is done by abstracting and parameterizing the templates. A limitation of OSTRICH, is that it currently
 75 only supports the top nine most used production-ready screen templates from OutSystems. The end-user may not
 76 create and save their own templates, nor can they re-apply a template which they have customized.

77 Another approach focused on enabling reuse of models, by converting and merging models into a single graph (the
 78 Knowledge Graph), which acts as a repository of models[24]. This graph is used to provide recommendations to the
 79 end-user, based on the model they're currently building. While this feature of recommending models (either constructed
 80 by domain experts and then developed by model experts, or made by the end-user themselves) could prove very useful,
 81 the study is clearly not focused on guiding the user towards reusing their own models.

82 Building on the ideas discussed for improving reuse in low-code development platforms (LCDPs), several popular
 83 tools show these concepts in action. For instance, Webflow[33] is a leading low-code platform that offers a wealth of
 84 features for building responsive websites. One of its standout features is the ability to create reusable components and
 85 UI kits, which can significantly speed up the development process. With Webflow's intuitive interface, developers can
 86 quickly design and prototype components, and then reuse them across multiple pages and projects. Despite all of the
 87 useful features that this tool has, it does not provide guidance to the end-users to create custom reusable components.

88 In a similar way, Mendix[39] takes this further for full enterprise apps by offering shareable building blocks like
 89 simple actions (microflows) and UI parts that anyone on a team can grab and use again without recoding. Through its
 90 Marketplace, a free online hub, you can download ready templates, connectors for tools like Salesforce, and basic setups
 91 that fit right into new projects, making everything faster and more uniform. This approach builds on the flexibility seen
 92 in platforms like Webflow, but adds strong team tools and AI suggestions to spot and create reusable pieces, empowering

even beginners to build complex apps while keeping reuse simple and widespread. This tool does offer guidance for the end-users to create custom reusable components through its AI suggestions, a lot of times these suggestions are not accurate enough (how do we know this??**).

OutSystems[40] further enhances the concept of reuse in low-code development platforms by emphasizing rapid application delivery through its robust set of features. Like Webflow and Mendix, OutSystems also provides a library of reusable components and templates that help developers complete projects faster. Its user-friendly visual development environment allows users to easily drag and drop elements while connecting with existing systems. OutSystems also supports teamwork with built-in version control and feedback features, making it easy for teams to share and improve reusable components. Additionally, the platform uses AI to suggest the best solutions and components for specific tasks. By encouraging reuse at both individual and team levels, OutSystems enables organizations to create scalable applications quickly while ensuring quality and consistency. Similarly to the previous tool explained, the AI suggestions that this tool provides are not always accurate to successfully guide the end-user to create custom reusable components (again, how do we know this??**).

In order to analyze how block-based robotics environments address reuse area, 4 representative platforms were compared: mBlock, MakeCode, SPIKE LEGO, VEXcode GO and Open Roberta. The comparison focused on three main dimensions of reuse: structural reuse (through user-defined blocks or functions), social reuse (through sharing or remixing existing projects), and interoperable reuse (through import/export capabilities).

Table 1. Block Based Robotics Environments Reuse Support

Platform	Structural Reuse	Social Reuse	Interoperable Reuse	Reuse Support
VEXcode GO	X	X		Medium
mBlock	X	X	X	Medium
MakeCode	X	X	X	Medium
Spike Lego	X		X	Low
Open Roberta		X		Low

In this context, “reuse support” represents a scale that measures how effectively each platform facilitates reuse-related features. High reuse support indicates that users can easily create, share, and adapt existing components or projects. Medium reuse support suggests that some reuse mechanisms are available but limited in scope or flexibility. Low reuse support implies that the platform provides only minimal or restricted features to promote reuse and improve user productivity.

As shown in Table 1, although these platforms include reusability features, they are quite limited, as none of them provide users with clear guidance on how to use these tools effectively, which restricts their ability to fully leverage them.

Lin and Weintrop (2021) noted that most existing research on block-based programming focuses on supporting the transition to text-based languages rather than exploring how features within BBP environments [31]—such as abstraction or reuse—can enhance learning outcomes. In contrast, our work emphasizes guided abstraction, helping users understand and practice modular design directly within block-based environments.

Techapalokul and Tilevich (2019) proposed extending the Scratch programming environment with facilities for reusing individual custom blocks to promote procedural abstraction and improve code quality. They observed that

157 while Scratch enables remixing of entire projects, it lacks mechanisms for reusing smaller, modular pieces of code. Their
 158 work suggests that supporting such fine-grained code reuse could enhance programmer productivity, creativity, and
 159 learning outcomes. Building on this idea, our project applies similar principles within the OpenRoberta environment
 160 by automating the detection of duplicate code segments and guiding users toward creating reusable custom blocks.
 161 Adler et al. (2021) introduced a search-based refactoring approach to improve the readability of Scratch programs by
 162 automatically applying small code transformations, such as simplifying control structures and splitting long scripts.
 163 Their findings demonstrated that automated refactoring can significantly enhance code quality and readability for
 164 novice programmers. Building upon this concept, our project applies similar principles in the OpenRoberta environment,
 165 focusing on detecting duplicate code segments and guiding users toward creating reusable custom blocks to promote
 166 modularity and abstraction.[3].
 167

168 Existing block-based environments provide mechanisms for reuse, but lack intelligent support to help users recognize
 169 and apply reuse in practice. To address this gap, our project introduces a guided reuse assistant within the Open Roberta
 170 Lab environment. The tool is designed to help users identify and apply reuse more easily while creating their robot
 171 programs. It works by automatically scanning a user's block-based program to detect repeated code segments in the
 172 workspace. The system visually highlights the found duplicates, drawing the user's attention to patterns that could be
 173 simplified.
 174

175 The tool also offers the functionality to create the custom block for the end-user, by identifying the small differences
 176 between the repeated parts—such as numbers, variables, or parameters—and turning these differences into inputs for
 177 the new block. The tool automatically replaces all relevant duplicate sequences with the new custom block.
 178

179 By combining ideas from procedural abstraction (organizing code into meaningful, reusable parts) and automated
 180 refactoring (improving code through intelligent transformations), our tool aims to make block-based programming
 181 more structured and efficient. It encourages users to build programs that are modular and easier to maintain, helps
 182 reduce unnecessary repetition, and supports learning by making the concept of reuse clear and hands-on.
 183

184 3 Study Design

185 3.1 Problem Investigation

186 3.1.1 *Problem Context and Motivation.* End-user development (EUD) for collaborative robots (cobots) presents unique
 187 challenges, particularly for users without formal programming training. In domains such as chemistry laboratories,
 188 educational robotics, and industrial settings, end-users need to program robots to perform specific tasks but often lack
 189 the software engineering knowledge to write maintainable, well-structured code. In the domain of Chemistry, one of
 190 the most prevalent and important tasks is performing experiments in labs in order to test a hypothesis, or to aid in the
 191 understanding of how chemicals react. Robots can be used in chemistry labs to automate experiments with great effect,
 192 as many experiments involve steps that are repetitive, and susceptible to human error - such as a step being overlooked,
 193 instructions being misread, etc. Automation of menial tasks will leave the chemists with more time for other work, and
 194 also comes with the added bonus of chemists not having to handle dangerous chemicals.
 195

196 One critical challenge in EUD is code reuse. Users frequently create repetitive code because they struggle to recognize
 197 duplicate patterns, lack knowledge about abstraction mechanisms, or find existing tools too complex to use effectively.
 198 This problem manifests in several ways: programs become unnecessarily long and difficult to maintain and small changes
 199 require modifications in multiple locations, increasing the risk of errors. Several visual programming environments,
 200 like OpenRoberta Lab, don't provide assistance in identifying when code should be reused or how to extract repeated
 201

209 sequences into reusable components. As lab work in chemistry involves many repetitive tasks, these challenges can
 210 easily become an obstacle for the chemists, which may turn them away from using cobots, as the inconvenience
 211 outweighs the benefits.
 212

213 **3.1.2 Stakeholder Analysis.**

- 214
- 215 • **Chemistry Laboratory Personnel:** Chemists and lab technicians who use cobots for repetitive tasks such as
 216 sample preparation, dispensing, mixing, and quality control procedures. They possess deep domain expertise in
 217 chemistry but limited programming knowledge, often creating long, repetitive programs that become difficult
 218 to maintain when adapting experimental protocols. Their primary need is to quickly create and modify robot
 219 programs without becoming programming experts.
 220

222 Table 2. Functional and Non-Functional Requirements
 223

Type	ID	Description	Priority
Functional	FR1	Detect duplicate/similar block sequences	High
	FR2	Visually highlight detected duplications	High
	FR3	Suggest creation of reusable custom blocks	High
	FR4	Allow users to accept/reject suggestions	High
Non-Functional	NFR1	Seamless Open Roberta Lab integration	High
	NFR2	Intuitive interface for end users	High
	NFR3	No interference with existing workflow	High
	NFR4	Clear visual feedback during detection	High

238 **3.1.3 Artifact Requirements.** The artifact requirements can be seen in table 2.
 239

240 **3.2 Treatment Design**
 241

242 Our treatment focuses on developing a guided reuse assistant for the OpenRoberta Lab environment. The purpose
 243 of this tool is to help users recognize which parts of their robot programs can be reused, and to make it easier for
 244 them to create reusable custom blocks. By doing this, we aim to reduce repetitive code and help users learn important
 245 programming concepts such as modularity and abstraction.
 246

247 **3.2.1 Overview of the Tool.** The guided reuse assistant is built as an extension inside Open Roberta Lab, which uses the
 248 Blockly framework. The assistant runs directly in the web browser and interacts with the user's block workspace. Its
 249 main job is to look through the user's program, find repeated sequences of blocks, and guide the user in turning them
 250 into reusable blocks.
 251

252 The tool works in three main steps:
 253

- 254 (1) **Detecting Repeated Code:** The assistant automatically scans the user's program and searches for parts that
 255 look the same or very similar. These are marked as potential duplicates.
 256
- 257 (2) **Highlighting and Suggesting Reuse:** Once duplicates are found, the system highlights them in the workspace
 258 and shows a message suggesting that these sections could be made into a reusable block (function). This helps
 259 users see repetition they might not have noticed before.
 260

- 261 (3) **Helping the User Create a New Block:** If the user agrees to the suggestion, the assistant opens a small guide
262 to help them create the new block. It automatically detects any small differences between the repeated parts,
263 such as numbers or variable names, and turns them into inputs (parameters) for the new block. When the block
264 is created, repeated code is replaced by the new reusable block.
265

3.3 Treatment Validation

3.3.1 Data Gathering and Analysis. The treatment validation for this study adopts a mixed-methods evaluation approach to assess the effectiveness of the proposed features for guiding users in creating reusable custom blocks within the OpenRoberta environment. Participants will be recruited from local educational institutions, specifically chemistry students and teachers who frequently engage in laboratory work. A sufficient number of (x) participants will be selected to ensure a diverse range of experience levels with block-based programming.

A pre-experiment survey/interview** will be used to gather data about the participants' demographic, and their understanding of modular programming concepts. This is followed by two tasks to be done in the OpenRoberta Lab, designed to make the user focus on reuse. The experimental setup will take place in a controlled environment, where participants will be divided into two groups: one using the enhanced OpenRoberta platform with guided block creation features, and the other using the standard version without these enhancements. Participants' interactions with the platform will be observed throughout the experiment. Data collection will include both quantitative measures, such as task completion time and accuracy in creating reusable blocks and qualitative feedback obtained through a post-task interview. For the qualitative feedback, both groups will have to repeat the task, with the group that initially used the enhanced OpenRoberta platform now using the standard version, while the other group will use the enhanced version. The analysis will compare performance metrics between the two groups and apply thematic analysis to the qualitative data to identify user experiences and perceptions of the new features' usability and effectiveness. This comprehensive evaluation will provide a detailed understanding of how useful and effective is the block creation guidance feature to the end-users.

3.3.2 Participant Recruitment. The participants will be chemistry students and one supervisor from the University of Southern Denmark (SDU). One of the authors of this paper knows a student from the chemistry line whom was recruited for the experiment. This student also assisted in recruiting others from his class. It should be noted, that this selection of participants classifies as a convenience sampling. As such, they may not represent the general population.

The participants will be asked to fill out a survey before starting the tasks, in order to assess their background, as well as their knowledge about block-based programming, the use of cobots, OpenRoberta Lab and their experience with programming. The survey can be found in appendix XXX.

3.3.3 Task Execution. Before the tasks, the participants will be given a short introduction to the OpenRoberta Lab, as well as the cobot simulator. The participants will then perform two tasks, each task described by a set of pre-defined steps to perform. The first task will be generic in nature. The purpose of this task is to make the user more familiar with block-based programming and the OpenRoberta Lab.

The second task is more focused on the domain of chemistry, as it is modelled after a real lab experiment performed by chemistry students at SDU (appendix XXX). The experiment instructions were obtained from one of the participants. The instructions for both tasks can be found in appendix XXX.

313 4 Results

314
315 The treatment validation concluded with 8 participants total. The results show that all participants preferred the
316 enhanced version of OpenRoberta Lab compared to the standard version, with 25% of participants finding the enhanced
317 version to be 'better' than the original, and 75% found it to be 'much better'. [insert piechart]. Results also showed that
318 75% of participants found the enhanced version 'easy' to use and 25% finding it 'very easy'.
319
320 Results showed a high level of satisfaction with the highlights, with 87,5% of users being 'satisfied' or 'very satisfied',
321 while 12,5% felt neutral about the highlights.

322 5 Discussion**323 5.1 Lessons Learned**

324
325 Based on the feedback from the participants, as well as observations of how they solved the task, the participants found
326 the enhanced version of OpenRoberta Lab to be better than the standard version. Noteably, 7 out of 8 participants
327 commented on how the enhanced version let them perform their task faster. As described in section 2, this is also one of
328 the main benefits of reuse in the field of software engineering. While a somewhat large(?) percentage of the participants
329 had no preference in regards to the visual look of the highlight, half of the users picked the 'Animated Color Highlight'.
330 This suggests that dynamic visuals - in this case: the blocks changing color repeatedly - are well-suited for catching the
331 user's attention.
332

333 Changes suggested by the participants mainly focus on smaller customizations of the tool and the OpenRoberta
334 Lab UI. It would be amiss to claim that the lack of suggested changes, focused on the tool overall, indicate that there
335 is no need for improvement of the tool. As many of the participants consider themselves 'beginners' in regards to
336 Computer Programming, it's likely that they lack ideas about other ways the tool could have been designed. Instead,
337 these answers can be interpreted as the participants having little to no issue with the current design.
338

339 5.2 Implications for Practice**340 5.3 Threats to Validity**

341 5.3.1 *Convenience Sampling*. The participants to the study were either acquaintances of one of the authors of the study,
342 or were recruited through these acquaintances. As such, the results of this study do not represent the general population
343 within the domain of chemistry.

344 5.3.2 *Limitations to observation*. Due to constraints with time and flexibility, only one of the authors was present to
345 observe the participants. To ensure that data from the observation was not affected by this, a screen recording of each
346 participant performing the task was saved. Several of the authors reviewed and discussed these recordings together to
347 extract data.

348 6 Modifications

349 Modifying the template – including but not limited to: adjusting margins, typeface sizes, line spacing, paragraph and
350 list definitions, and the use of the \vspace command to manually adjust the vertical spacing between elements of your
351 work – is not allowed.
352

353 **Your document will be returned to you for revision if modifications are discovered.**

365 7 Typefaces

366 The “acmart” document class requires the use of the “Libertine” typeface family. Your TeX installation should include
 367 this set of packages. Please do not substitute other typefaces. The “lmodern” and “ltimes” packages should not be used,
 368 as they will override the built-in typeface families.
 369

371 8 Title Information

373 The title of your work should use capital letters appropriately - <https://capitalizemytitle.com/> has useful rules for
 374 capitalization. Use the `title` command to define the title of your work. If your work has a subtitle, define it with the
 375 `subtitle` command. Do not insert line breaks in your title.
 376

377 If your title is lengthy, you must define a short version to be used in the page headers, to prevent overlapping text.
 378 The `title` command has a “short title” parameter:
 379

```
380    \title[short title]{full title}
```

382 9 Authors and Affiliations

384 Each author must be defined separately for accurate metadata identification. As an exception, multiple authors may
 385 share one affiliation. Authors’ names should not be abbreviated; use full first names wherever possible. Include authors’
 386 e-mail addresses whenever possible.
 387

388 Grouping authors’ names or e-mail addresses, or providing an “e-mail alias,” as shown below, is not acceptable:

```
389    \author{Brooke Aster, David Mehldau}
  390    \email{dave,judy,steve@university.edu}
  391    \email{firstname.lastname@phillips.org}
```

393 The `authornote` and `authornotemark` commands allow a note to apply to multiple authors – for example, if the
 394 first two authors of an article contributed equally to the work.
 395

396 If your author list is lengthy, you must define a shortened version of the list of authors to be used in the page headers,
 397 to prevent overlapping text. The following command should be placed just after the last `\author{}` definition:
 398

```
399    \renewcommand{\shortauthors}{McCartney, et al.}
```

401 Omitting this command will force the use of a concatenated list of all of the authors’ names, which may result in
 402 overlapping text in the page headers.
 403

404 The article template’s documentation, available at <https://www.acm.org/publications/proceedings-template>, has a
 405 complete explanation of these commands and tips for their effective use.
 406

407 Note that authors’ addresses are mandatory for journal articles.
 408

408 10 Rights Information

410 Authors of any work published by ACM will need to complete a rights form. Depending on the kind of work, and the
 411 rights management choice made by the author, this may be copyright transfer, permission, license, or an OA (open
 412 access) agreement.
 413

414 Regardless of the rights management choice, the author will receive a copy of the completed rights form once it
 415 has been submitted. This form contains L^AT_EX commands that must be copied into the source document. When the
 416 Manuscript submitted to ACM

417 document source is compiled, these commands and their parameters add formatted text to several areas of the final
418 document:

- 419
- 420 • the “ACM Reference Format” text on the first page.
 - 421 • the “rights management” text on the first page.
 - 422 • the conference information in the page header(s).

423 Rights information is unique to the work; if you are preparing several works for an event, make sure to use the
424 correct set of commands with each of the works.

425 The ACM Reference Format text is required for all articles over one page in length, and is optional for one-page
426 articles (abstracts).

427 **11 CCS Concepts and User-Defined Keywords**

428 Two elements of the “acmart” document class provide powerful taxonomic tools for you to help readers find your work
429 in an online search.

430 The ACM Computing Classification System — <https://www.acm.org/publications/class-2012> — is a set of classifiers
431 and concepts that describe the computing discipline. Authors can select entries from this classification system, via
432 <https://dl.acm.org/ccs/ccs.cfm>, and generate the commands to be included in the L^AT_EX source.

433 User-defined keywords are a comma-separated list of words and phrases of the authors’ choosing, providing a more
434 flexible way of describing the research being presented.

435 CCS concepts and user-defined keywords are required for all articles over two pages in length, and are optional
436 for one- and two-page articles (or abstracts).

437 **12 Sectioning Commands**

438 Your work should use standard L^AT_EX sectioning commands: \section, \subsection, \subsubsection, \paragraph,
439 and \ subparagraph. The sectioning levels up to \subsubsection should be numbered; do not remove the numbering
440 from the commands.

441 Simulating a sectioning command by setting the first word or words of a paragraph in boldface or italicized text is
442 **not allowed**.

443 Below are examples of sectioning commands.

444 **12.1 Subsection**

445 This is a subsection.

446 *12.1.1 Subsubsection.* This is a subsubsection.

447 *Paragraph.* This is a paragraph.

448 Subparagraph This is a subparagraph.

449 **13 Tables**

450 The “acmart” document class includes the “booktabs” package — <https://ctan.org/pkg/booktabs> — for preparing
451 high-quality tables.

452 Table captions are placed *above* the table.

Table 3. Frequency of Special Characters

Non-English or Math	Frequency	Comments
\emptyset	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

Table 4. Some Typical Commands

Command	A Number	Comments
\author	100	Author
\table	300	For tables
\table*	400	For wider tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper “floating” placement of tables, use the environment **table** to enclose the table’s contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material are found in the *L^AT_EX User’s Guide*.

Immediately following this sentence is the point at which Table 3 is included in the input file; compare the placement of the table here with the table in the printed output of this document.

To set a wider table, which takes up the whole width of the page’s live area, use the environment **table*** to enclose the table’s contents and the table caption. As with a single-column table, this wide table will “float” to a location deemed more desirable. Immediately following this sentence is the point at which Table 4 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed output of this document.

Always use midrule to separate table header rows from data rows, and use it only for this purpose. This enables assistive technologies to recognise table headers and support their users in navigating tables more easily.

14 Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

14.1 Inline (In-text) Equations

A formula that appears in the running text is called an inline or in-text formula. It is produced by the **math** environment, which can be invoked with the usual `\begin{math} . . . \end{math}` construction or with the short form `$. . . $`. You can use any of the symbols and structures, from α to ω , available in L^AT_EX [29]; this section will simply show a few examples of in-text equations in context. Notice how this equation: $\lim_{n \rightarrow \infty} x = 0$, set here in in-line math style, looks slightly different when set in display style. (See next section).

521 14.2 Display Equations

522 A numbered display equation—one set off by vertical space from the text and centered horizontally—is produced by the
 523 **equation** environment. An unnumbered display equation is produced by the **displaymath** environment.

524 Again, in either environment, you can use any of the symbols and structures available in L^AT_EX; this section will just
 525 give a couple of examples of display equations in context. First, consider the equation, shown as an inline equation
 526 above:

$$529 \lim_{n \rightarrow \infty} x = 0 \quad (1)$$

530 Notice how it is formatted somewhat differently in the **displaymath** environment. Now, we'll enter an unnumbered
 531 equation:

$$533 \sum_{i=0}^{\infty} x + 1$$

535 and follow it with another numbered equation:

$$537 \sum_{i=0}^{\infty} x_i = \int_0^{\pi+2} f \quad (2)$$

539 just to demonstrate L^AT_EX's able handling of numbering.

540 15 Figures

543 The “figure” environment should be used for figures. One or more images can be placed within a figure. If your figure
 544 contains third-party material, you must clearly identify it as such, as shown in the example below.

545 Your figures should contain a caption which describes the figure to the reader.

546 Figure captions are placed *below* the figure.

547 Every figure should also have a figure description unless it is purely decorative. These descriptions convey what's in
 548 the image to someone who cannot see it. They are also used by search engine crawlers for indexing images, and when
 549 images cannot be loaded.

550 A figure description must be unformatted plain text less than 2000 characters long (including spaces). **Figure**
 551 **descriptions should not repeat the figure caption – their purpose is to capture important information that is**
 552 **not already provided in the caption or the main text of the paper.** For figures that convey important and complex
 553 new information, a short text description may not be adequate. More complex alternative descriptions can be placed in
 554 an appendix and referenced in a short figure description. For example, provide a data table capturing the information in
 555 a bar chart, or a structured list representing a graph. For additional information regarding how best to write figure
 556 descriptions and why doing this is so important, please see <https://www.acm.org/publications/taps/describing-figures/>.

557 15.1 The “Teaser Figure”

558 A “teaser figure” is an image, or set of images in one figure, that are placed after all author and affiliation information,
 559 and before the body of the article, spanning the page. If you wish to have such a figure in your article, place the
 560 command immediately before the `\maketitle` command:

```
561 \begin{teaserfigure}
562   \includegraphics[width=\textwidth]{sampleteaser}
563   \caption{figure caption}
564   \Description{figure description}
```



Fig. 1. 1907 Franklin Model D roadster. Photograph by Harris & Ewing, Inc. [Public domain], via Wikimedia Commons. (<https://goo.gl/VLCRBB>).

```
609 \end{teaserfigure}
```

611 **16 Citations and Bibliographies**

613 The use of BibTeX for the preparation and formatting of one’s references is strongly recommended. Authors’ names
 614 should be complete – use full first names (“Donald E. Knuth”) not initials (“D. E. Knuth”) – and the salient identifying
 615 features of a reference should be included: title, year, volume, number, pages, article DOI, etc.
 616

617 The bibliography is included in your source document with these two commands, placed just before the \end{document}
 618 command:

```
620 \bibliographystyle{ACM-Reference-Format}
621 \bibliography{bibfile}
```

622 where “*bibfile*” is the name, without the “.bib” suffix, of the BibTeX file.

623 Manuscript submitted to ACM

625 Citations and references are numbered by default. A small number of ACM publications have citations and references
 626 formatted in the “author year” style; for these exceptions, please include this command in the **preamble** (before the
 627 command “`\begin{document}`”) of your L^AT_EX source:

629 `\citetstyle{acmauthoryear}`

631 Some examples. A paginated journal article [2], an enumerated journal article [14], a reference to an entire issue [13],
 632 a monograph (whole book) [28], a monograph/whole book in a series (see 2a in spec. document) [21], a divisible-book
 633 such as an anthology or compilation [16] followed by the same example, however we only output the series if the volume
 634 number is given [17] (so Editor00a’s series should NOT be present since it has no vol. no.), a chapter in a divisible book
 635 [46], a chapter in a divisible book in a series [15], a multi-volume work as book [27], a couple of articles in a proceedings
 636 (of a conference, symposium, workshop for example) (paginated proceedings article) [4, 19], a proceedings article with
 637 all possible elements [45], an example of an enumerated proceedings article [18], an informally published work [20], a
 638 couple of preprints [7, 11], a doctoral dissertation [12], a master’s thesis: [5], an online document / world wide web
 639 resource [1, 36, 47], a video game (Case 1) [35] and (Case 2) [34] and [30] and (Case 3) a patent [44], work accepted for
 640 publication [41], ‘YYYYb’-test for prolific author [42] and [43]. Other cites might contain ‘duplicate’ DOI and URLs
 641 (some SIAM articles) [26]. Boris / Barbara Beeton: multi-volume works as books [23] and [22]. A presentation [38]. An
 642 article under review [8]. A couple of citations with DOIs: [25, 26]. Online citations: [47–49]. Artifacts: [37] and [6].
 643

644 17 Acknowledgments

645 Identification of funding sources and other support, and thanks to individuals and groups that assisted in the research
 646 and the preparation of the work should be included in an acknowledgment section, which is placed just before the
 647 reference section in your document.

648 This section has a special environment:

649 `\begin{acks}`
 650 ...
 651 `\end{acks}`

652 so that the information contained therein can be more easily collected during the article metadata extraction phase, and
 653 to ensure consistency in the spelling of the section heading.

654 Authors should not prepare this section as a numbered or unnumbered `\section`; please use the “`acks`” environment.

665 18 Appendices

666 If your work needs an appendix, add it before the “`\end{document}`” command at the conclusion of your source
 667 document.

668 Start the appendix with the “`appendix`” command:

669 `\appendix`

670 and note that in the appendix, sections are lettered, not numbered. This document has two appendices, demonstrating
 671 the section and subsection identification method.

677 19 Multi-language papers

678 Papers may be written in languages other than English or include titles, subtitles, keywords and abstracts in different
 679 languages (as a rule, a paper in a language other than English should include an English title and an English abstract).
 680 Use `language=...` for every language used in the paper. The last language indicated is the main language of the paper.
 681 For example, a French paper with additional titles and abstracts in English and German may start with the following
 682 command
 683

```
684 \documentclass[sigconf, language=english, language=german,
685   language=french]{acmart}
```

686 The title, subtitle, keywords and abstract will be typeset in the main language of the paper. The commands
 687 `\translatedXXX, XXX` begin title, subtitle and keywords, can be used to set these elements in the other languages. The
 688 environment `translatedabstract` is used to set the translation of the abstract. These commands and environment have
 689 a mandatory first argument: the language of the second argument. See `sample-sigconf-i13n.tex` file for examples of
 690 their usage.
 691

692 20 SIGCHI Extended Abstracts

693 The “sigchi-a” template style (available only in L^AT_EX and not in Word) produces a landscape-orientation formatted
 694 article, with a wide left margin. Three environments are available for use with the “sigchi-a” template style, and
 695 produce formatted output in the margin:
 696

700 **sidebar:** Place formatted text in the margin.

701 **marginfigure:** Place a figure in the margin.

702 **margintable:** Place a table in the margin.

703 Acknowledgments

704 To Robert, for the bagels and explaining CMYK and color spaces.

705 References

- 706 [1] Rafal Ablamowicz and Bertfried Fauser. 2007. *CLIFFORD: a Maple 11 Package for Clifford Algebra Computations, version 11*. Retrieved February 28,
 707 2008 from <http://math.tntech.edu/rafal/cliff11/index.html>
- 708 [2] Patricia S. Abril and Robert Plant. 2007. The patent holder’s dilemma: Buy, sell, or troll? *Commun. ACM* 50, 1 (Jan. 2007), 36–44. doi:10.1145/
 709 1188913.1188915
- 710 [3] Felix Adler, Gordon Fraser, Eva Gründinger, Nina Körber, Simon Labrenz, Jonas Lerchenberger, Stephan Lukasczyk, and Sebastian Schweikl. 2021.
 711 Improving Readability of Scratch Programs with Search-Based Refactoring. In *Proceedings of the IEEE/ACM 43rd International Conference on Software
 712 Engineering: Companion Proceedings (ICSE-SEET)*. IEEE. doi:10.1109/ICSE-Companion.2021.00105
- 713 [4] Sten Andler. 1979. Predicate Path expressions. In *Proceedings of the 6th ACM SIGACT-SIGPLAN symposium on Principles of Programming Languages
 714 (POPL ’79)*. ACM Press, New York, NY, 226–236. doi:10.1145/567752.567774
- 715 [5] David A. Anisi. 2003. *Optimal Motion Control of a Ground Vehicle*. Master’s thesis. Royal Institute of Technology (KTH), Stockholm, Sweden.
- 716 [6] Sam Anzaroot and Andrew McCallum. 2013. *UMass Citation Field Extraction Dataset*. Retrieved May 27, 2019 from [http://www.iesl.cs.umass.edu/data/data-umasscitationfield](http://www.iesl.cs.umass.edu/

 717 data/data-umasscitationfield)
- 718 [7] Sam Anzaroot, Alexandre Passos, David Belanger, and Andrew McCallum. 2014. *Learning Soft Linear Constraints with Application to Citation Field
 719 Extraction*. arXiv:1403.1349 doi:10.48550/arXiv.1403.1349
- 720 [8] R. Baggett, M. Simecek, C. Chambellan, K. Tsui, and M. Fraune. 2025. Fluidity in the Phased Framework of Technology Acceptance: Case Study to
 721 Gain a Holistic Understanding of (Older Adult) Participant Advancement Through Acceptance Phases with Mobile Telepresence Robots. *Robotics
 722 Aut. Systems*. Manuscript submitted for review.
- 723 [9] Len Bass, Paul Clements, and Rick Kazman. 2021. *Software Architecture in Practice, 4th Edition*. Addison-Wesley Professional.
- 724 [10] Alexander Bock and Ulrich Frank. 2021. Low-Code Platform. *Business and Information Systems Engineering* 63 (2021). doi:10.1007/s12599-021-00726-8

725 Manuscript submitted to ACM

- [11] Lutz Bornmann, K. Brad Wray, and Robin Haunschild. 2019. *Citation concept analysis (CCA)—A new form of citation analysis revealing the usefulness of concepts for other researchers illustrated by two exemplary case studies including classic books by Thomas S. Kuhn and Karl R. Popper*. arXiv:1905.12410 [cs.DL]
- [12] Kenneth L. Clarkson. 1985. *Algorithms for Closest-Point Problems (Computational Geometry)*. Ph.D. Dissertation. Stanford University, Palo Alto, CA. UMI Order Number: AAT 8506171.
- [13] Jacques Cohen (Ed.). 1996. Special issue: Digital Libraries. *Commun. ACM* 39, 11 (Nov. 1996).
- [14] Sarah Cohen, Werner Nutt, and Yehoshua Sagiv. 2007. Deciding equivalences among conjunctive aggregate queries. *J. ACM* 54, 2, Article 5 (April 2007), 50 pages. doi:10.1145/1219092.1219093
- [15] Bruce P. Douglass, David Harel, and Mark B. Trakhtenbrot. 1998. Statecharts in use: structured analysis and object-orientation. In *Lectures on Embedded Systems*, Grzegorz Rozenberg and Frits W. Vaandrager (Eds.). Lecture Notes in Computer Science, Vol. 1494. Springer-Verlag, London, 368–394. doi:10.1007/3-540-65193-4_29
- [16] Ian Editor (Ed.). 2007. *The title of book one* (1st. ed.). The name of the series one, Vol. 9. University of Chicago Press, Chicago, Chapter The title of the chapter, 127–238. doi:10.1007/3-540-09237-4
- [17] Ian Editor (Ed.). 2008. *The title of book two* (2nd. ed.). University of Chicago Press, Chicago, Chapter 100, 25–137. doi:10.1007/3-540-09237-4
- [18] Matthew Van Gundy, Davide Balzarotti, and Giovanni Vigna. 2007. Catch me, if you can: Evading network signatures with web-based polymorphic worms. In *Proceedings of the first USENIX workshop on Offensive Technologies (WOOT '07)*. USENIX Association, Berkley, CA, Article 7, 9 pages.
- [19] Torben Hagerup, Kurt Mehlhorn, and J. Ian Munro. 1993. Maintaining Discrete Probability Distributions Optimally. In *Proceedings of the 20th International Colloquium on Automata, Languages and Programming (Lecture Notes in Computer Science, Vol. 700)*. Springer-Verlag, Berlin, 253–264.
- [20] David Harel. 1978. *LOGICS of Programs: AXIOMATICS and DESCRIPTIVE POWER*. MIT Research Lab Technical Report TR-200. Massachusetts Institute of Technology, Cambridge, MA.
- [21] David Harel. 1979. *First-Order Dynamic Logic*. Lecture Notes in Computer Science, Vol. 68. Springer-Verlag, New York, NY. doi:10.1007/3-540-09237-4
- [22] Lars Hörmander. 1985. *The analysis of linear partial differential operators. III*. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Vol. 275. Springer-Verlag, Berlin, Germany. viii+525 pages. Pseudodifferential operators.
- [23] Lars Hörmander. 1985. *The analysis of linear partial differential operators. IV*. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Vol. 275. Springer-Verlag, Berlin, Germany. vii+352 pages. Fourier integral operators.
- [24] Ilirian Ibrahim and Dimitris Moudilos. 2022. Towards model reuse in low-code development platforms based on knowledge graphs. In *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings* (Montreal, Quebec, Canada) (MODELS '22). Association for Computing Machinery, New York, NY, USA, 826–836. doi:10.1145/3550356.3561570
- [25] IEEE 2004. IEEE TCSC Executive Committee. In *Proceedings of the IEEE International Conference on Web Services (ICWS '04)*. IEEE Computer Society, Washington, DC, USA, 21–22. doi:10.1109/ICWS.2004.64
- [26] Markus Kirschmer and John Voight. 2010. Algorithmic Enumeration of Ideal Classes for Quaternion Orders. *SIAM J. Comput.* 39, 5 (Jan. 2010), 1714–1747. doi:10.1137/080734467
- [27] Donald E. Knuth. 1997. *The Art of Computer Programming*, Vol. 1: *Fundamental Algorithms* (3rd. ed.). Addison Wesley Longman Publishing Co., Inc., Boston.
- [28] David Kosuri. 2001. *Understanding Policy-Based Networking* (2nd. ed.). Wiley, New York, NY.
- [29] Leslie Lamport. 1986. *LaTeX: A Document Preparation System*. Addison-Wesley, Reading, MA.
- [30] Newton Lee. 2005. Interview with Bill Kinder: January 13, 2005. Video. *Comput. Entertain.* 3, 1, Article 4 (Jan.-March 2005). doi:10.1145/1057270.1057278
- [31] Yuhan Lin and David Weintrop. 2021. The Landscape of Block-Based Programming: Characteristics of Block-Based Environments and How They Support the Transition to Text-Based Programming. *Journal of Computer Languages* 67 (2021), 101075. doi:10.1016/j.cola.2021.101075
- [32] Hugo Lourenço, Carla Ferreira, and João Costa Seco. 2021. OSTRICH - A Type-Safe Template Language for Low-Code Development. In *2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. 216–226. doi:10.1109/MODELS50736.2021.00030
- [33] Vlad Magdalin. 2012. Low code platform tool Webflow. <https://webflow.com/>.
- [34] Dave Novak. 2003. Solder man. Video. In *ACM SIGGRAPH 2003 Video Review on Animation theater Program: Part I - Vol. 145 (July 27–27, 2003)*. ACM Press, New York, NY, 4. doi:10.945/woot07-S422 <http://video.google.com/videoplay?docid=6528042696351994555>
- [35] Barack Obama. 2008. A more perfect union. Video. Retrieved March 21, 2008 from <http://video.google.com/videoplay?docid=6528042696351994555>
- [36] Poker-Edge.Com. 2006. Stats and Analysis. Retrieved June 7, 2006 from <http://www.poker-edge.com/stats.php>
- [37] R Core Team. 2019. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. <https://www.R-project.org/>
- [38] Brian J. Reiser. 2014. Designing coherent storylines aligned with NGSS for the K-12 classroom. Presentation at National Science Education Leadership Association Meeting, Boston, MA, USA. <https://www.academia.edu/6884962/>
- [39] Derek Roos. 2005. Low code platform tool Mendix. <https://www.mendix.com/>.
- [40] Paulo Rosado. 2011. Low code platform tool Outsystems. <https://www.outsystems.com/>.
- [41] Bernard Rous. 2008. The Enabling of Digital Libraries. *Digital Libraries* 12, 3, Article 5 (July 2008). To appear.
- [42] Mehdi Saeedi, Morteza Saheb Zamani, and Mehdi Sedighi. 2010. A library-based synthesis methodology for reversible logic. *Microelectron. J.* 41, 4 (April 2010), 185–194.

- [781] [43] Mehdi Saeedi, Morteza Saheb Zamani, Mehdi Sedighi, and Zahra Sasanian. 2010. Synthesis of Reversible Circuit Using Cycle-Based Approach. *J. Emerg. Technol. Comput. Syst.* 6, 4 (Dec. 2010), 12 pages.
- [782] [44] Joseph Scientist. 2009. The fountain of youth. Patent No. 12345, Filed July 1st., 2008, Issued Aug. 9th., 2009.
- [783] [45] Stan W. Smith. 2010. An experiment in bibliographic mark-up: Parsing metadata for XML export. In *Proceedings of the 3rd. annual workshop on Librarians and Computers (LAC '10, Vol. 3)*, Reginald N. Smythe and Alexander Noble (Eds.). Paparazzi Press, Milan Italy, 422–431.
- [784] [46] Asad Z. Spector. 1990. Achieving application requirements. In *Distributed Systems* (2nd. ed.), Sape Mullender (Ed.). ACM Press, New York, NY, 19–33. doi:10.1145/90417.90738
- [785] [47] Harry Thornburg. 2001. *Introduction to Bayesian Statistics*. Retrieved March 2, 2005 from <http://ccrma.stanford.edu/~jos/bayes/bayes.html>, archived at [<https://web.archive.org/web/20240505055615/https://ccrma.stanford.edu/~jos/bayes/bayes.html>]
- [786] [48] TUG 2017. *Institutional members of the TeX Users Group*. Retrieved May 27, 2017 from <http://www.tug.org/instmem.html>
- [787] [49] Boris Veytsman. 2017. *acmart—Class for typesetting publications of ACM*. Retrieved May 27, 2017 from <http://www.ctan.org/pkg/acmart>

A Research Methods

A.1 Part One

Lore ipsum dolor sit amet, consectetur adipiscing elit. Morbi malesuada, quam in pulvinar varius, metus nunc fermentum urna, id sollicitudin purus odio sit amet enim. Aliquam ullamcorper eu ipsum vel mollis. Curabitur quis dictum nisl. Phasellus vel semper risus, et lacinia dolor. Integer ultricies commodo sem nec semper.

A.2 Part Two

Etiam commodo feugiat nisl pulvinar pellentesque. Etiam auctor sodales ligula, non varius nibh pulvinar semper. Suspendisse nec lectus non ipsum convallis congue hendrerit vitae sapien. Donec at laoreet eros. Vivamus non purus placerat, scelerisque diam eu, cursus ante. Etiam aliquam tortor auctor efficitur mattis.

B Online Resources

Nam id fermentum dui. Suspendisse sagittis tortor a nulla mollis, in pulvinar ex pretium. Sed interdum orci quis metus euismod, et sagittis enim maximus. Vestibulum gravida massa ut felis suscipit congue. Quisque mattis elit a risus ultrices commodo venenatis eget dui. Etiam sagittis eleifend elementum.

Nam interdum magna at lectus dignissim, ac dignissim lorem rhoncus. Maecenas eu arcu ac neque placerat aliquam. Nunc pulvinar massa et mattis lacinia.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009