Algorithmic Trading Systems and Strategies: A New Approach

Design, Build, and Maintain an Effective Strategy Search Mechanism

Viktoria Dolzhenko

Apress®

Algorithmic Trading Systems and Strategies: A New Approach

Design, Build, and Maintain an Effective Strategy Search Mechanism

Viktoria Dolzhenko

Apress®

Algorithmic Trading Systems and Strategies: A New Approach: Design, Build, and Maintain an Effective Strategy Search Mechanism

Viktoria Dolzhenko
San Jose, Costa Rica

ISBN-13 (pbk): 979-8-8688-0356-7 ISBN-13 (electronic): 979-8-8688-0357-4
https://doi.org/10.1007/979-8-8688-0357-4

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Malini Rajendran
Development Editor: James Markham
Coordinating Editor: Gryffin Winkler

Cover designed by eStudioCalamar

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Apress Media, LLC, 1 New York Plaza, New York, NY 10004, U.S.A. Phone 1-800-SPRINGER, fax (201) 348-4505, email orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub (https://github.com/Apress). For more detailed information, please visit https://www.apress.com/gp/services/source-code.

If disposing of this product, please recycle the paper

This book is dedicated to all those who are constantly in search of new ideas. These tireless adventurers are undeterred by any difficulties or failures and have been looking for something new all their life to improve both their lives and the lives of the people around them. I admire such people immensely. It is thanks to them that the world continues to improve at least a little bit.

Dare and create. This is the purpose of life.

Table of Contents

About the Author xi

About the Technical Reviewers xiii

Introduction xv

Chapter 1: Popular Approaches to Developing Trading Systems 1

- Manual Trading 2
- Ready-Made Signals and Algorithms 2
  - Signals 3
  - Third-Party Algorithms 5
- Specialized Services 9
- Independent Creation of a Trading Platform 10
  - Testing a Single Strategy 10
  - Various Developer Approaches 12
- Hiring Third-Party Developers 12
- My Approach 13
- Summary 15

Chapter 2: Introduction to Developing Trading Systems 17

- General Theory 18
  - Order Execution 21
  - Margin and Leverage 24
- Composition of the Trading System 25
  - Trading Theory 27

Capital Management 39
Risk Control 49
Testing 58
Performance Indicators 59
Optimization 63
Summary 67

Chapter 3: Architectural Solution Part 1: Identifying the Requirements 69

Requirements Elicitation 71
Signals 72
First View of the Process 74
Theory Generator 78
Strategies Searching 82
Selection and Forward Testing 97
Selection of Financial Instruments 98
Setting Up a Search for a Profitable Strategy 99
The Logic of Searching for Profitable Strategies 100
Real Trading 101
Important Questions 103
Life Cycle of a Position 103
Capital Management 106
Risk Control 107
Scalability of Indicators 114
Summary 116

Chapter 4: Architectural Solution Part 2: Services and Subsystems 117

Microservice Architecture 118

Kubernetes 122

Subsystems 124

Strategy Search Subsystem 125

Processing Generators 126

Queue 130

Finite State Machine 134

Concept of Theory Processing Steps 135

Calculation of Subtheories 140

Core 149

Sandbox Exchange 150

Real Trading Subsystem 151

Integration with Exchanges 152

Launch and Operation of Strategies 156

Enabling and Disabling Strategies 158

Checking the Type of Financial Instrument 160

Master Data 167

Summary 169

Chapter 5: Technology Stack and Libraries 171

Choosing a Framework 172

Application Architecture 172

Spaghetti Code 173

Clean Architecture 174

Domain-Driven Design vs. Anemic Model 177

Object Relational Mapper 178
How to Use Dapper 179
Migrations 184
Finite State Machine 185
Principle 186
Hosted Service 189
Backworker 207
Summary 213

Chapter 6: Optimization Algorithms 215

Formulation of the Problem 216
Population Algorithms 217
Genetic Algorithms 220
Mutation Operators 222
Crossover Operators 224
Filtering Operators 228
Selection Operators 234
Restrictions 238
Local Unconstrained Optimization Algorithms 249
Summary 264

Chapter 7: Implementation of Optimization Algorithms 265

General View 266
Brute-Force Algorithm 270
Getting Info 272
Getting a Set of Values 277
How to Use 282
Genetic Algorithm 285
Steps 286

Getting Info 287
Getting a Set of Values 294
Initialization Step 303
Mutation Step 306
Filtering Step 309
Breeding Step 314
Test Functions 321
SubTheory Example 322
Summary 325

Chapter 8: Implementation of the Core Module 327

Use Cases 327
Context 331
Update Candle Event 332
Check Signals 335
Strategy Model 337
Calculate Signal 343
Calculation of Indicators 351
Average True Range (Atr) 357
Process of Positions 362
Process Bot (Lite Version) 364
Process Steps 373
Events 376
Process Acts 390
Summary 396

Chapter 9: Approaches to Final Implementation 397

Binance Adapter.....397

Implementation 398

Docker.....403

History 403

Why Is This Needed? 405

Docker Components 406

Launching the Application 408

Kubernetes.....415

Components 416

Pods.....417

Deployments.....419

Services.....424

Helm 426

Summary.....429

Index.....431

About the Author

Viktoria Dolzhenko My dream of becoming a programmer started the moment a computer first appeared in my home. As a child I loved programming and started regularly participating in programming competitions. And now, I have more than 10 years of experience in developing complex systems from scratch.

My professional passion is the complex design of various applications, while personally I am interested in algorithmic trading. To bring these two ideas together, I decided to build a system that would search for and find profitable strategies on its own. At that time, developing such a system for algorithmic trading was an interesting and difficult task.

Now I am sharing my findings and conclusions in this book so that everyone who wants to build their own algorithmic systems will not waste time creating ineffective systems. Readers can learn from my experience to gain the maximum benefits from their creations.

About the Technical Reviewers

Ganesh Harke is a seasoned technology leader working with a multinational investment bank. Over the past 14 years, Ganesh has channeled his expertise in design, architecture, and development of scalable and low-latency systems into various industries.

In his current role as technical lead at Citibank, Ganesh is at the forefront of developing a cutting-edge e-commerce platform. His professional journey extends beyond Citibank, with previous impactful roles at esteemed organizations such as Barclays and Siemens. He has contributed to and thrived in dynamic environments at each juncture, leaving an indelible mark on projects ranging from intricate financial solutions to cutting-edge technological innovations.

Ganesh has a master's degree in financial engineering and a bachelor's degree in information technology and has passed all three CFA levels.

Ganesh is also a mentor who helps fellow professionals and colleagues with his technical expertise. Ganesh was recognized for his work inside and outside of his career network.

Vadzim Zylevich is a certified Microsoft Solutions Developer with a strong background in software development, focusing on technologies like C# and ASP.NET. Over his 10-year career, he has led several important projects and has been recognized for his technical skills and leadership. Currently, he works at Maersk, the leading global logistics company, where he continues to apply his expertise to innovate and improve their technological capabilities. Beyond programming, he is a technical writer at Code-Maze and mentors aspiring developers, sharing his knowledge and experience. His interest in the impact of AI on software development led him to discuss this topic in a TickerNews interview, highlighting his commitment to both innovation and education in the tech field. His goal is to not only advance in technology but also support the growth of the tech community by making complex concepts accessible to all.

As a senior portfolio performance and risk expert, **Samantha** delivers multi-asset expertise within her area and clarifies the strategic solutions to global asset managers and sell-side players leveraging Bloomberg's suite of solutions and models. Specializing in liquidity stress assessment, fixed-income analytics, and customized indexing within the context of various investment vehicles, Samantha has a wealth of knowledge of the ever-shifting investment landscape and the relevant world-class solutions to address the regulatory and industry challenges.

Introduction

If you start researching algorithmic trading, you will notice a general pattern in the logic of creating trading systems. That pattern is to find a few high-profit strategies and use them in the trading. Often the search is carried out using a lot of manual labor. There is one big drawback in this approach: there is a low probability that a profitable strategy will be found.

Also, because of the amount of manual labor in this process, excessive demands are placed on the strategies found. They require a lot of support and configuration.

I remember when I tried many different strategies, most of which showed low performance. I was increasingly frustrated with myself and the approaches I was using. One day, I remembered some work I did with a teacher at my university. We were looking for optimal parameters for launching and adjusting the movement of artificial earth satellites to launch them into the target orbit with the least amount of fuel consumed. I thought, how does this task differ from the task of finding a profitable strategy? I realized that they were the same, which meant my knowledge in that area could be applied in a completely new direction.

As a result, I combined my knowledge of creating high-load systems and the field of engineering to create a new approach for finding and using profitable strategies. The solution lies not in looking for dozens of highly profitable strategies. The solution is to quickly find hundreds of strategies, albeit with more modest results, with minimal manual labor.

In this book, I not only describe a system that continuously searches for and runs strategies but also show an option for implementing the fundamental parts of such a system using the .NET Framework. In this book you will learn how you can build a trading system that finds profitable strategies with a higher probability than with conventional approaches.

CHAPTER 1

Popular Approaches to Developing Trading Systems

Before you start creating your own system, you must first look for possible ways to implement it. Perhaps someone has already implemented your idea and there is no need to create your own program. Even if there is nothing like your idea on the market, you can still glean valuable insights from existing products.

Currently, there are many approaches to creating trading systems, ranging from using conventional manual trading systems on the stock exchange to hiring third-party developers or a company to implement your ideas. In fact, you can not only find ideas for trading systems on the Internet, but also find their implementations.

In this chapter, you will learn about the main approaches to creating trading systems and consider their advantages and disadvantages. I will also describe my approach and explain why I chose it.

V. Dolzhenko, *Algorithmic Trading Systems and Strategies: A New Approach*,
https://doi.org/10.1007/979-8-8688-0357-4_1

Manual Trading

The most famous traders earned their fortunes through their knowledge, effort, and vast experience. If you want to follow their path, you need to hone your trading skills and increase your knowledge every day.

The essence of a trading system based on manual trading is an independent search for successful strategies, gaining experience from daily failures and successes. I am fascinated by successful traders who do not use third-party assistant programs to trade. Naturally, these people use classic charts provided by brokers, but they do the main work of analyzing their strategies on their own. No matter how popular artificial neural networks are, they do not yet have intuition and subconsciousness—an undeniable advantage of humans.

However, it is worth noting that this plus is also a minus. Emotions and greed have long plagued humanity, and traders are no exception. Often, when an important decision needs to be made, a person succumbs to the crowd effect, or even panic, and thoughtlessly takes on unnecessary risk. That is why, in manual trading, human error is high. Mistakes, in turn, lead to undermining the human nervous system.

Emotions will never get the best of a trading robot. In addition, it is capable of making decisions much faster than a human, and it does not require rest or sleep; therefore, the number of trading transactions it can complete is significantly greater than that of a human.

Ready-Made Signals and Algorithms

The easiest trading system to understand and implement can be based either on trading signals from other trading participants or on the signals or automatic trading of a trading robot that you have purchased. Both of these approaches can generate income, but only if used correctly. Don't be duped by what their sellers promise. That is, you need to use them as tools and not all-encompassing solutions.

Signals

Signals in trading are hints or recommendations from a *signal provider* to make purchases and sales of financial instruments. That is, a signal tells you to buy or sell. In practice, if this signal is in the form an alert, it could look like a short text message, as shown here:

TSLA. Buy price 203, Sell price 208

Signal providers can be both ordinary traders and serious experienced analysts at specialized companies. There are a huge number of signal providers on the market; the main problem is finding a high-quality and honest company.

Signals are convenient because they require absolutely no time to develop and test. When you buy one, you receive a ready-made solution; all that is required is to follow the instructions. But as they say, the devil is in the details. Despite the apparent simplicity of the signals, suppliers often hide or provide partial information about the logic of the decision-making involved, so users do not know exactly what logic the signal is based on. Therefore, to decide whether to work with a certain provider and its signal, you need to carefully study its analytics.

Signals come in different types depending on the control and price. These divisions are somewhat superficial, but here are the distinctions:

- **Hand signals.** These are signals that the supplier provides using manual trading. They are often the result of sitting in front of a monitor for a long time and studying the instrument for a long time. Most likely, there will not be very many of these signals from one supplier.
- **Automatic signals.** These are signals based on the trading algorithm of the supplier. That is, these signals are generated automatically. Naturally, with this approach, the number of generated signals will most likely be higher than with manual signals.

- **Copy trading.** When copying a trade, the seller's account and the buyer's account are linked, and when the seller's account opens a transaction, it is automatically opened in the buyer's account. This is a simple principle that allows the buyer not to spend a lot of time trading. But the biggest disadvantage of this case is the lack of sufficient control over your account.
- **Alerts.** These are signals in the form of text messages that tell the user what to do. I showed an example of an alert earlier. This is one of the safest types of signals, since the decision for action and position size always remains with the account owner.
- **Account management.** With this option, the client completely transfers control of their account to the supplier. For this, the supplier will most likely take a certain percentage of the profit. When choosing this option, you should carefully read the information about the company providing such services. Also, there is a high risk of fraud here.
- **Free.** Free signals are generally of lower quality than ones that cost money. This is logical as almost no one wants to share the results of their work for free.

At first glance, the task of creating a trading system based on ready-made signals looks simple. The user creates a portfolio of such signals from different suppliers and works on it. The main difficulty here is that it is necessary to spend quite a lot of time on analytics and on testing such signals, constantly adding new ones and eliminating outdated ones. This means that the relative simplicity of creating such a system results in enormous difficulties and costs to maintain it.

Third-Party Algorithms

The global trend to automate and algorithmize everything has now come to trading, and every year the number of algorithmic systems that help traders or that even trade independently is growing. Anyone can buy an algorithmic robot; you don't need to be a programmer.

There are so many of trading robots that the question becomes, why haven't the people who created them used them to make themselves millions? Some may have, of course, but ironically it's not so much thanks to trading, but rather thanks to the proceeds from selling the robots to consumers.

My point is that you need to understand that no one would be selling a trading robot if trading with it could bring in more money than the sales from the robot itself. This does not mean that there are no helpful robots, just that robots can only do so much with search and analytics.

You should also pay attention to the rather high price of robots. That is, you'll want to get a profitable trading algorithm but also earn back the money you invested in the robot. In fact, finding a decent robot that will work consistently is quite difficult. At the time of writing this book, one of the popular sites for selling algorithmic robots sells about 4,000 "expert" robots. Imagine how hard it is to analyze them all and make sure they work! You should not only be looking at the reviews but also analyzing the indicators yourself before buying one.

That said, anyone who does not want to create all this from scratch can easily buy a robot. In fact, some people have hundreds of such robots in their portfolio and yet they don't spend a second of their time creating them.

There are different types of algorithms used in robots. The types differ not only in the logic of decision-making but also in money management, average trading time, trading method, price, and many other parameters. In addition, any trading robot must contain the analytical data of its work, which is valuable information for the buyer.

The following is a short list of indicators that can be used to analyze a robot's performance; I will talk about them in more detail in the next chapter:

- **Expected value.** This is an estimate of the average expected return of a strategy during its long-term use. Essentially, it tells you how successful a strategy can be over the long term.
- **Profit factor.** This is an indicator of the effectiveness of a trading strategy, which is the ratio of the amount of profit to the amount of losses. This shows how your profits compare to your losses.
- **Absolute drawdown.** This represents the change in funds in a trading account from the beginning to the end of the trading period.
- **Relative drawdown.** This drawdown is the trader's largest loss as a percentage relative to the previous maximum balance.
- **Maximum drawdown.** This shows the maximum decrease in capital from the highest level it has ever been to the lowest it has ever been.
- **Recovery factor.** The recovery factor is equal to the absolute value of net profit divided by the maximum drawdown. The higher the recovery factor, the faster the system recovers after a drawdown.

As you can understand, there are many types of robots. The following are some of them:

- **Long-term, medium-term, short-term.** This is a classification based on the average time a position is held.

- **Short-term ones, in turn, are divided into arbitrage and scalping.** Arbitrage robots open positions not on one exchange but on several, but their positions are always short-term and last literally milliseconds. In this they are similar to scalping ones, but the latter trade on only one exchange.
- **Automatic and semi-automatic.** While automatic systems carry out activities without human assistance, semi-automatic ones require constant attention. This is because semi-automatic systems have much less functionality and are replaced by human labor. For example, such a system may require information about the size of the position being opened.
- **Expert advisors based on the Martingale principle.** This is a money management system in which after each unprofitable closing of a position, it is necessary to double the size of the next position. If the outcome is profitable, you can cover losses from the previous transaction and return to the original position size.
- **Trend and oscillator.** These are systems based on technical analysis.
- **Nonindicator.** These are specific algorithmic robots. Examples are systems that make decisions based on the analysis of news sites or forums.

For short-term signals, one of the most important analytical indicators is the average time of holding a position. After all, if this indicator is equal to or less than one minute, this means you are dealing with scalper or arbitrage signals. These signals are extremely sensitive to ping, and most likely you will not have a chance to trade them in time. Therefore, I advise you to consider signals only with a holding position of at least several

minutes or more. Lengthy deals are also a bad sign. Perhaps the robot is trying to wait out the accumulated losses on transactions and the decision-making logic contains a serious flaw, reflected in an untimely exit from the position.

In addition, you should absolutely be scared off by Martingale advisors. Despite its apparent simplicity, this system will almost always lead you to losing all your capital. With a long series of losing trades, your capital will be reset to zero. It is quite easy to determine whether a system belongs to Martingale by looking at the balance chart and the funds chart. These graphs for normal systems should be as close as possible and naturally tend to grow. A uniform alternation of open and closed positions also indicates the reliability of the system. But large gaps between charts and an increase in the number of open positions and a decrease in closed ones indicates Martingale.

Automatic robots are generally considered more reliable than semi-automated ones. After all, the human factor is completely excluded from them. But it should also be noted that they are much more complex in development than semi-automatic ones, and the price for the highest quality ones is, accordingly, higher. It is also worth noting the need for total control over the operation of such a system so that you or the monitoring system can detect a problem in a timely manner and stop trading.

The market for algorithmic robots is dominated by trend and oscillatory systems. And this is logical, because the formulas of technical indicators can be easily transferred into the program. They are easy to see on stock charts and easy to analyze.

The undeniable advantage of any algorithmic robot is its ability to trade different instruments. In fact, this feature alone suggests that the robot is quite reliable; it is obvious that this robot does not work by adjustment, which means that it will give good results in the long term and respond well to market changes.

Someone who wants to create their own trading system based on purchasing algorithms can take the path of diversification and risk reduction. Buy a large number of robots for your portfolio, preferably working on different principles, and use them comprehensively. In essence, the principle is similar to creating a portfolio from trading signals. Here, too, most of the time will be spent on constant analysis and testing. But this approach, unfortunately, is more expensive.

Specialized Services

All the previous options have one drawback—they do not implement your ideas for trading strategies exactly. Of course, you can create your own trading system using your own ideas. But what do you do if you have no programming skills? In this case, special services can come to the rescue, where it is possible, without any special knowledge, to assemble your robot step-by-step. You can simply implement the decision-making logic and then test and analyze the resulting strategy. Some services even allow you to set up the money management rules and risk control, and others allow you to complicate the logic using a special scripting language.

Unfortunately, it's difficult to create something specific with these types of services. For more advanced strategies, you need to learn at least an easy scripting language to go beyond simple logic. Some services provide programming capabilities using languages such as Python or C#, both of which allow you to create more complex algorithms. In addition, there are many more such services other than "constructor" services. They usually have a large number of users and thus a developed learning environment.

Difficulties arise when you want to create something unique to test a theory that does not fit into the standard framework. This is the time to switch to independent development or to hire third-party developers.

Independent Creation of a Trading Platform

Having your own trading system has a big advantage over other approaches—you have the ability to test any of your theories since you are not limited by the functionality included in a ready-made solution. The prospects for this direction are enormous, because they are not limited in any way, by anyone, or by anything. The process is surprisingly creative and unique to each developer. Moreover, the developer is not tied to any specific programming language; you can use the technology stack that you know. Most likely, once you choose this course, you will never switch to another, simpler one.

The division into different approaches in this category is arbitrary. After all, adding or removing a functionality to/from a program changes the system every time, and it becomes completely different. But still, let's look at some different approaches so that you have an idea of what suits you best.

Testing a Single Strategy

Let's imagine that you have a brilliant idea and want to test it. You will quickly program the logic and test it through your own or some third-party service. Everything that your trading system will contain is a single strategy. At the same time, the functionality of your trading platform is not important; it may or may not contain a testing or optimization module. All your platform does is trade according to a single given scenario.

The undeniable advantage of this approach is its high development speed. You don't need to worry about the optimal value of your strategy parameters or the impact of any indicator on performance.

The obvious disadvantage is the lack of understanding of what can be improved in the strategy or how the performance would change if you added or removed some condition or changed some indicator parameter.

Roughly speaking, the main disadvantage is a lack of searching for the optimal strategy for your theory. But there is a theory in your strategy; you just immediately set certain parameters for it.

Here is an example of this strategy.

Input signal:

BBW > 0.05

WMA < Open Price

Output signal:

BBW < 0.01

RSI > 30

You can, of course, evaluate this strategy in terms of performance indicators, but it would be much better to consider how these indicators change if the specific parameters of the strategy also changed. The brute-force method will be the easiest to implement here.

The brute-force method, although ancient and elementary, has always helped humanity in incomprehensible situations. It can also serve an excellent purpose for a simple developer. If you implement the brute-force method into the previous strategy, you can see how the specific value of each parameter affects performance. That is, by setting a specific range and step, you can build a graph of changes in any performance indicator depending on changes in the parameter.

The brute-force method does an excellent job of analyzing the influence of specific values of indicator parameters on the effectiveness of the theory. Thus, by sequentially or comprehensively going through all the possible options for parameters, you can conduct an accurate analysis of the theory. But this method is effective only on a small number of parameters. If the theory contains a large number of possible strategy options, then finding the optimal value will take an incredibly large amount of time.

Various Developer Approaches

Basically, those who really take the matter seriously develop entire trading platforms with a lot of functionality. Some write their own analog of the exchange to have their own independent testing. Optimization modules, modules for searching for optimal strategies, real trading module, money management module, and much more. Some write their own analog of the exchange to have their own independent testingTrading system developmentindependent testing, optimization modules, modules for searching for optimal strategies, real trading module, money management module, and much more.

The theories that cannot be verified by standard auxiliary services can be verified in your own system. For example, say you want to test the market psychological theory that the stock price of a certain company directly depends on mentions of this company on social networks. You can create a parser that could analyze the mood of the masses using hashtags and generate signals when a critical mass is reached in a positive or negative direction. Or you have the idea that some stocks are directly dependent on other stocks. Or that a certain company's stock goes up when it rains. Crazy? Maybe, but you can test all your ideas when you do independent development.

Hiring Third-Party Developers

At some point, a mechanical trading trader may decide to automate their system, in whole or in part, which will allow them to free up more time for deeper analysis and improvement of trading strategies and theories. However, if the trader has no programming experience, this can be intimidating. After all, no one wants to spend months or even years studying this when it's unclear whether the time spent will pay off.

Therefore, despite that there are many services on the market that help you create a full-fledged trading robot, it is still a very labor-intensive

task. As a result, it is often the successful traders who consider hiring an experienced engineer to design and develop their own trading system. That is, when someone has a great theory but neither the time nor the opportunity to implement the idea programmatically, they often hire third-party developers.

The first disadvantage of this approach is the need to share your profitable idea with other people. This may lead to the emergence of competitors who then start working according to your theory. So, you need to protect your intellectual property. A special contract may help with this, but it still may not help. In any case, you definitely shouldn't trust the developers' word. It may seem right to share only part of your idea with the developer, but in this case, the developed system may not work as you originally planned. Therefore, if you decide to hire programmers, then it is better to draw up the detailed technical specifications yourself.

The second disadvantage is that it is unclear whether you will recoup your development costs. You spend a lot of money to bring your theory to life, but it might not lead to any profit.

Another important question is where to find a good developer for your system. There are also many paths here. The route of hiring a third-party company seems to be the most suitable, and most likely your intellectual property will remain yours. After all, a company's reputation is worth a lot. But be prepared to spend more money than if you hired a freelance developer.

My Approach

Each of the previously discussed approaches has its place, and each of them is capable of bringing profit to the trader. In this book, I will describe my approach and how I created a trading platform without hiring third-party developers, without using specialized programs, and so on.

In the beginning, I analyzed the existing approaches of developing trading robots, studied the robots for sale at that time, read blogs of successful practicing algorithmic traders, and so on. As a result, I realized almost all traders were looking for super-profitable strategies. It had to be a strategy that brought 1,000 percent per month and had a drawdown of 1 percent and no more. Of course, each trader had their own concept of an extremely profitable strategy, but most of them would not even look at a strategy with a very modest result of 3 percent per year.

I didn't initially understand this concept. After all, even if a person is very smart, there is a possibility that they will not be able to find such a strategy even if they spend their whole life searching. I understood that the likelihood that I would find a super-profitable strategy was very low. So, I decided to try the diversification approach. I thought, what if I create strategies that are fundamentally different, with modest indicators, for example, 0.5 to 1 percent per month, but use them in real trading in hundreds? Most likely, there are many more such strategies than super-profitable ones, and, naturally, they are much easier to find.

As a result, I chose the option of completely independent development with the idea of automatically searching for many profitable strategies, but with modest indicators. Of course, if the system finds a super-profitable strategy, then good; if it doesn't find it, it's OK. Thus, I wanted to create a system that every developer could do independently and that could almost 100 percent of the time generate additional income. Yes, it may not be millions of dollars, but it will work consistently and will require low operating costs.

The big disadvantage of this approach is that it takes a huge amount of time to implement. In fact, this process will never end. I constantly want to improve something or add some functionality. But this is a very creative and interesting process that I advise for everyone.

Summary

In this chapter, I briefly discussed the types of trading systems and how to create them, and I discussed their advantages and disadvantages.

- **Manual trading.** The main advantage of this approach is the complete absence of labor costs for creating a trading system, as well as the use of human intelligence and experience. Among the shortcomings, the most striking are instability due to the emotional state of a person, as well as a relatively low number of open positions.
- **Trading using ready-made signals.** The advantage of this approach is that there is no need to implement a trading system. The main disadvantage is that you must trust the experience of others, with a possible lack of understanding of the logic of how the signals work.
- **Trading using purchased algorithmic robots.** Again, the undeniable advantage of this approach is that there is no need to create a trading system. You will be working using someone else's system. This also entails disadvantages: you will understand the real capabilities of system only *after* the purchase. And often you will misunderstand the internal logic of such systems.
- **Independent development of individual strategies.** This is a popular approach primarily because of its simplicity and flexibility. You can implement your own unique strategy, but speed comes at the cost of limited functionality.

- **Hiring third-party developers.** With this approach you can get almost anything, including implementing my approach in developing a trading system. The main disadvantages are the difficulty of finding good developers, the need to share your ideas, and the high cost.
- **Independent development of a trading system.** With this approach, you either independently or in a circle of a small number of partners independently implement not a single strategy but an entire system with automatic search, testing, and launch strategies. The obvious advantage of this idea is the flexibility of strategy implementation. The big disadvantage of this approach is the large labor costs for implementing such a system.